НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»
Теплоенергетичний факультет
Кафедра автоматизації проектування енергетичних процесів і систем

До захисту допущено:
Завідувач кафедри
_____ Олександр КОВАЛЬ
«___»_____2020 р.

Дипломна робота

на здобуття ступеня бакалавра
за освітньо-професійною програмою «Інформаційні технології моніторингу довкілля»
спеціальності 122 «Комп'ютерні науки та інформаційні технології»
на тему: «Інструментальні засоби локалізації об'єктів на основі технології біконів**»**

Виконала:
студентка IV курсу, групи ТМ-62
Лебедик Тетяна Олександрівна _____
Керівник:
доцент, кандидат економічних наук
Гусєва Ірина Ігорівна _____
Рецензент:
доцент, кандидат технічних наук
Веремійчука Юрій Андрійович _____

Засвідчую, що у цій дипломній роботі немає запозичень з праць інших авторів без відповідних посилань.
Студентка _____

Київ – 2020 року

Національний технічний університет України
“Київський політехнічний інститут імені Ігоря Сікорського”

Факультет теплоенергетичний
Кафедра автоматизації проектування енергетичних процесів і систем
Рівень вищої освіти перший рівень
Напрям підготовки 122 Комп’ютерні науки та інформаційні технології
Спеціалізація Інформаційні технології моніторингу довкілля

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____– Олександр КовальКОВАЛЬ
— (підпис)

”___”_____2020р.

ЗАВДАННЯ
на дипломну роботу студенту

_____Лебедик Тетяні Олександрівні_____
(прізвище, ім’я, по батькові)

1. Тема роботи Інструментальні засоби локалізації об’єктів на основі технології біконів_____

керівник роботи_Гусєва Ірина Ігорівна, доцент, кандидат економічних наук__
(прізвище, ім’я, по батькові науковий ступінь, вчене звання)

затверджена наказом вищого навчального закладу від ”25” травня 2020р. № **1168-с**

2. Строк подання студентом роботи ___12.06.20_____
3. Вихідні дані до роботи _Kotlin, SQL, Android_____
4.Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) Провести аналіз предметної області, а саме: сферу та методи використання біконів, а також облік відвідування у навчальних закладах. Розробити базу даних та мобільні додатки для керування нею. Виконати тестування системи. Проаналізувати результати апробації та удосконалити програмні продукти._____
5. Перелік ілюстративного матеріалу
__“Тема”, “Локалізація обєктів”, “Мета розробки”, “Технологія BLE”, “Бікони”, “Основні поля пакетів”, “Засоби розробки”, “Як працює система?”, “Концептуальна модель бази даних”, “Діагарама прецедентів систми”, “Мобільний додаток викладача -1”, “Мобільний додаток викладача-2”, “Мобільний додаток студента”, “Веб-додаток адміністратора-1”, “Веб-додаток адміністратора-2”, “Висновки”_____

6. Дата видачі завдання ”_11_”___жовтня___2020 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів виконання дипломної роботи	Термін виконання етапів роботи	Примітки
1.	Затвердження теми роботи	8.02.20	
2.	Вивчення та аналіз задачі	13-19.04.20	
3.	Розробка архітектури та загальної структури системи	20-26.04.20	
4.	Розробка структур окремих підсистем	27.04-03.05.20	
5.	Програмна реалізація системи	04-13.05.20	
6.	Оформлення пояснювальної записки	14-16.05.20	
7.	Захист програмного продукту	17.05.20	
8.	Передзахист	12.06.20	
9.	Захист	15.06.20	

Студент _____ _____Тетяна ЛЕБЕДИК~~Лебедик Т.О.~~_____

(підпис) (ім'я та прізвище~~ та ініціали,~~)

Керівник роботи _____ _____Ірина ГУСЄВА~~Гусєва І.І.~~_____

(підпис) (ім'я та прізвище~~ та ініціали,~~)

АНОТАЦІЯ

Дипломна робота на тему “Інструментальні засоби локалізації об’єктів на основі технології біконів”.

Метою роботи є створення програмного забезпечення, яке буде вирішувати проблему обліку відвідування навчального закладу, на основі технології біконів.

В роботі розглянуті теоретичні та практичні аспекти розробки програмного продукту для вирішення проблеми обліку відвідування навчального закладу. Спроектовано базу даних, розроблено два мобільні додатки для викладача та студента, а також веб-додаток для адміністрування.

Пояснювальна записка містить 59 сторінок, включає 33 рисунки, 11 таблиць, 3 додатки і список використаних джерел з 23 найменувань.

Ключові слова:

Система обліку відвідування, програмне забезпечення, технологія Bluetooth, бікони, мобільна розробка, Android.

ABSTRACT

Thesis on "Tools for localization of objects based on beacon technology."

The purpose of the work is to create software that will solve the problem of accounting for school attendance. It is based on beacon technology.

The paper considers theoretical and practical aspects of software product development to solve the problem of accounting for school attendance. A database has been designed, two mobile applications for teachers and students and a web application for administration have been developed.

The explanatory note contains 59 pages, includes 33 pictures, 11 tables, 3 appendices and the list of the used sources from 23 names.

Keywords:

Attendance accounting system, software, Bluetooth technology, beacons, mobile development, Android.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ ТА СКОРОЧЕНЬ

Wi-Fi – технологія бездротової локальної мережі

Bluetooth – виробнича специфікація бездротових персональних мереж

BLE – технологія цифрової бездротової передачі даних з наднизьким енергоспоживанням

AA – «пальчикова батарейка», типорозмір гальванічних елементів

USB – універсальна послідовна шина, призначена для з'єднання комп'ютерів і периферійних пристроїв

IEEE - Інститут інженерів з електротехніки та електроніки

JVM – віртуальна машина Java

СУБД – система управління базами даних

IOT – інтернет речей

CRUD – чотири базові функції управління даними “створення, зчитування, зміна і видалення”

ICM – група радіочастотних діапазонів

ГГц – гігагерц

БД – база даних

ВСТУП

Сьогодні гостро постає питання локалізації об’єктів на місцевості. Вже важко знайти людину, котра б не використовувала мобільні додатки для навігації. Кожен день мільйони мешканців нашої планети вмикають свої мобільні телефони, щоб розглянути карти та знайти потрібну інформацію, пов’язану з місцем їх локації.

Для цих цілей на відкритій місцевості давно використовується Система глобального позиціювання (англ. Global Positioning System). Проте, для внутрішньої локалізації об’єктів, на вулицях міста чи в середині будівель, вона є неефективною в наслідок ослаблення сигналу та його загасання.

Останні роки для вирішення проблеми внутрішньої локалізації вивчають та впроваджують декілька основних технологій: Wi-Fi та Bluetooth.

Встановлення точок доступу Wi-Fi, на перший погляд, найбільш логічне рішення, проте, є декілька ключових недоліків даної технології. По-перше, вона має обмежену кількість точок доступу та має багато незручностей у їх впровадженні. По-друге, Wi-Fi здебільшого використовують для покриття певної території сигналом, а не для вирішення питань локалізації. Отже, проблему внутрішньої локалізації об’єктів вона не вирішує.

Альтернативним та найбільш оптимальним варіантом є використання Bluetooth технології. А точніше, технології, яка виникла на її основі – Bluetooth з низьким енергоспоживанням (англ. Bluetooth Low Energy, або скорочено BLE).

BLE – технологія, яка дозволяє транслювати дані широкомовним сигналом, використовуючи мінімальну кількість енергії. Технологія має низьку виробничу вартість, легкодоступна для користувачів системи і дуже проста у розгортанні. Технологія широко досліджувалася і вчені прийшли до висновків, що точність використання її у декілька разів вища, ніж при використанні конкуруючих технологій[1]. Таким чином вона є ідеальною для впровадження, як частина системи внутрішньої локалізації об’єктів.

Bluetooth з низьким енергоспоживанням використовує бікони (англ. beacons), для забезпечення внутрішньої навігації.

Бікон - це пристрій достатньо малого розміру, що транслює в широкомовному форматі пакети з даними через невеликі інтервали часу. Незважаючи на малі розміри, завдяки низькому та оптимізованому процесу використання енергії, вони можуть автономно працювати протягом достатньо довгого періоду часу. Бікон в пакетах, що ним транслюються, передає інформацію про сам бікон та певні показники, що зазвичай використовуються для обчислення відстані. Мобільні пристрої з Bluetooth зчитують ці пакети і дані, що знаходяться в них, можуть бути використані програмами, що встановлені на мобільному пристрої[2]. Існує багато протоколів, які використовуються для роботи з біконами. Проте найбільш широко використовуються наступні:

- iBeacon – створений компанією Apple
- Eddystone – протокол, розроблений Google
- AltBeacon – представлений RadiusNetworks

Детальніше структуру протоколів буде розглянуто далі.

Сьогодні, бікони, з кожним днем, стають більш популярними, за рахунок їх невеликої вартості, зручності впровадження та налагодження . Все більше компаній впроваджують системи внутрішньої локалізації на основі BLE технології.

Основною метою роботи є створення програмного забезпечення, на основі технології біконів, яке буде вирішувати важливу проблеми в галузі знань, а саме проблему обліку відвідування навчального закладу. Автоматизація даного процесу значно полегшить роботу викладачам та зекономить багато часу, що допоможе оптимізувати навчальний процес.

1. ЗАДАЧА ЛОКАЛІЗАЦІЇ ОБ'ЄКТІВ НА ОСНОВІ ТЕХНОЛОГІЇ БІКОНІВ

Мета розробки: створити програмне забезпечення, яке буде вирішувати проблему обліку відвідування навчального закладу, на основі технології біконів.

Задачі:

1. Створити мобільний додаток для викладача.
2. Створити мобільний додаток для студента.
3. Створити базу даних для для збереження інформації.
4. Створити веб-додаток для адміністрування бази даних.

Користувачі системи: викладачі, студенти та адміністратори бази даних

Вхідні дані системи: розклад, дані про викладачів, дані про студентів, дані про бікони

Вихідні дані системи: таблиці відвідування

Детальний опис та постановка задач для кожного підмодулю системи виглядає наступним чином:

Мобільний додаток студента

Створити мобільний додаток, який би надав можливість студенту:

- переглянути список предметів на сьогодні
- переглянути список викладачів
- переглянути інформацію про викладача
- переглянути розклад викладача
- переглянути мапу навчального закладу
- поставити відмітку про присутність на парі

Мобільний додаток викладача

Створити мобільний додаток, який би надав можливість викладачу:

- переглянути список предметів на сьогодні
- переглянути список студентів

- переглянути інформацію про студента
- переглянути відвідування по групах
- переглянути інформацію про присутність студента на парі

Веб-додаток для адміністрування

- CRUD операції з таблицями бази даних

2. МЕТОДИ ТА ЗАСОБИ ЛОКАЛІЗАЦІЇ ОБ’ЄКТІВ

2.1 Існуючі технології локалізації об’єктів

Зараз широко впроваджуються та використовуються технології, що забезпечують бездротове з’єднання різних пристроїв. Однією з таких технологій є Bluetooth.

Специфікацію Bluetooth було розроблено групою Bluetooth Special Interest Group, скорочено Bluetooth SIG. Bluetooth SIG – всесвітня організація зі стандартів та стандартування, яка сьогодні здійснює контроль за розробкою стандартів бездротового з’єднання Bluetooth, а також займається ліцензуванням усіх виробників технологій, які його використовують за основу або як частину технічної реалізації.

До складу групи Bluetooth Special Interest Group входять такі компанії як Ericsson, IBM, Intel, Toshiba, а також Nokia. Згодом організація змогла досягти угоди з IEEE, й специфікація Bluetooth офіційно стала частиною усім відомого стандарту IEEE 802.15.1.

Bluetooth є однією з основних на сьогодні бездротових технологій, яка має надзвичайно високі темпи розвитку і дуже швидко впроваджується в різні галузі, як наприклад електроніка, телекомунікації, споживча галузь та інші[4].

BLE є свого роду IOT-версією специфікації Bluetooth. Технологія споживає аж у 15-25 разів менше електроенергії ніж її старша версія і саме завдяки низькому споживанню енергії, технологія BLE ідеально підходить для пристроїв, що залежать від акумулятора, незалежно від його типу.

Системи BLE працюють у без ліцензійному діапазоні ICM 2,4 ГГц і використовують стрибки частоти для мінімізації перешкод та колізій від інших радіочастотних пристроїв, які працюють в тій же смузі (наприклад, Wi-Fi або класичний Bluetooth)[5].

Для взаємодії між пристроями, що працюють з технологією BLE, один з них повинен взяти на себе роль рекламодавця, а інша - роль сканера.

У 2019 році було прогнозовано розгортання приблизно шістдесяти мільйонів біконів, насправді ця цифра виявилася навіть більша. Ринок технології BLE до 2024 року, за прогнозами зросте до 25 мільярдів доларів.

Технологія BLE з кожним роком значно розширює використання в промисловості та роздрібній торгівлі. Завдяки її швидкому розгортанню, все більше людей дізнається про неї, та починає впроваджувати та використовувати.

BLE має високий потенціал у сфері розумних будинків з малою потужністю, низькими витратами та невеликими пристроями. Бікони вже використовуються у сфері IOT, проте їх використання може бути розширене[2]. Наразі проводиться безліч досліджень та готуються нові моделі використання.

Ось лише мала доля варіантів використання, що вже впроваджено:

- реклама, проведення акцій та впровадження системи знижок у сфері торгівлі
- навігація та представлення інформації про експонати в музеях
- навігація на конференціях та інших масових заходах
- локалізація та навігація в середині будівель та відстеження внутрішніх переміщень
- системи допомоги людям з обмеженими можливостями
- економія енергії для розумного офісу
- управління розумними будинками і складами

Сьогодні більшість варіантів використання біконів зосереджена впровадженні на статичних об’єктах. Наприклад, навігація на вулицях міста чи в будівлях. Проте, динаміка розвитку технології уже дозволяє впроваджувати її на рухомих об’єктах, що відкриває нові можливості[3].

Деякі варіанти використання BLE технології у сфері IOT візуалізовано на

рисунку 2.1. Вся потрібна для системи інформація зберігається у хмарному сховищі і завдяки бікону в потрібний час і в потрібному місці транслюється користувачу системи.

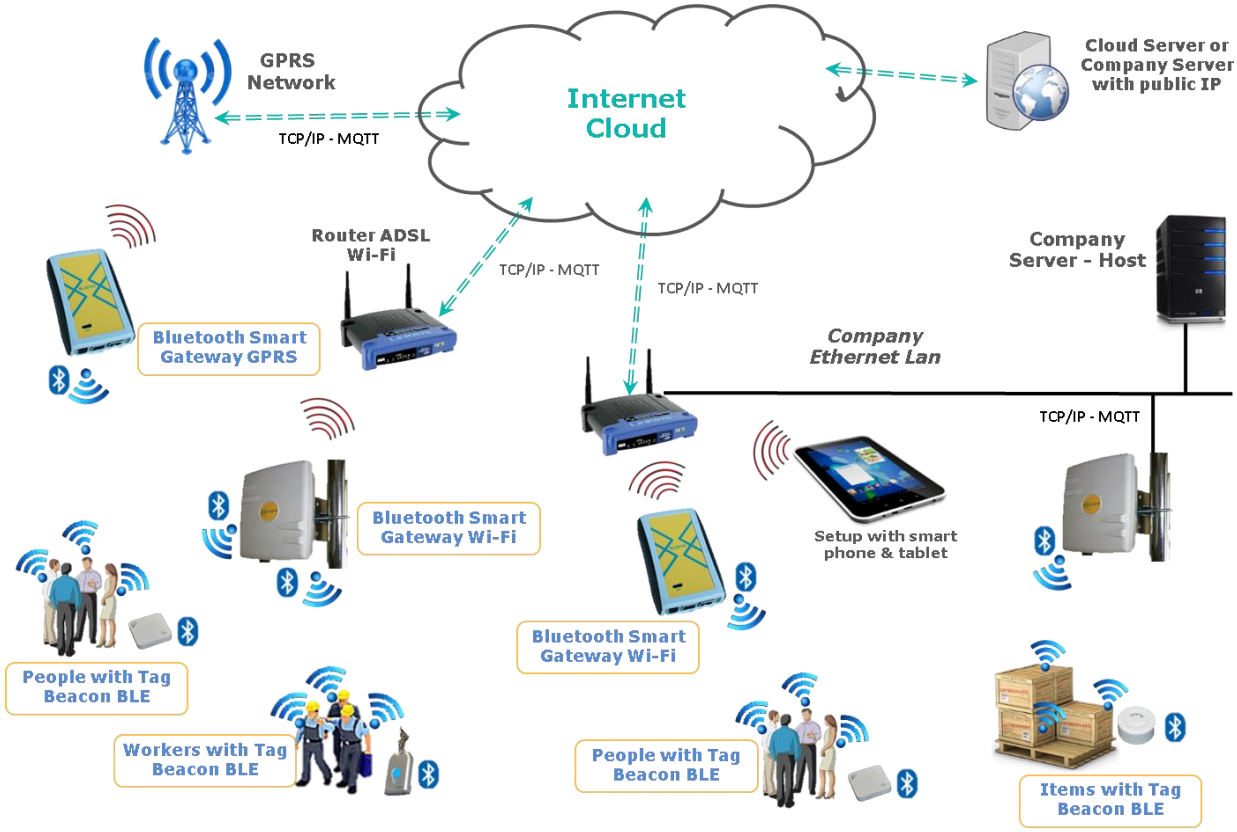

Рис. 2.1 – Варіанти використання біконів у сфері IOT систем[6].

Використання біконів на автомобілях, поїздах, велосипедах та людях вже широко практикується та показує гарні результати, проте ще потребує досліджень в сфері надійності та зонування.

Технологія Bluetooth з низьким енергоспоживанням є достатньо дешевою фінансово та легкою у впровадженні, що значно підвищує її шанси на лідерство, поміж аналогічних технологій у фінансовій та маркетинговій галузі.

Також потенційно успішними бікони можуть стати у автотранспортній та дорожній сфері. Контрольно-пропускні пункти, контроль руху, системи сповіщення – далеко не кінцевий список можливих варіантів використання.

З впровадженням біконів на автотранспорті, він перестане бути відокремленим один від одного і зможе працювати, як єдиний організм. А це

означає, що транспорт та його програмне забезпечення зможе реагувати на будь-які несподівані ситуації миттєво й відразу зможе, наприклад, перебудувати заданий маршрут, враховуючи нові обставини на дорозі.

Таким чином, технологія з кожним днем знаходить нові варіанти використання, а отже й підвищує свою актуальність. Тому, її використання для системи обліку відвідування навчального закладу є ідеальним варіантом для локалізації об’єктів.

2.2 Технологія біконів

Бікони - це пристрої BLE, які періодично транслюють невеликі пакети даних, працюють на монетних елементах живлення. Також доступні варіанти, що використовують, такі батареї як AA та USB. Завдяки технології BLE, бікони можуть безперебійно працювати місяцями чи навіть роками. Окрім цього, завдяки малому розміру бікони легко переносити та встановлювати[2].

Існує безліч варіантів та типів біконів. За способом живлення їх розділяють два основні типи:

- бікони, що живляться від батарей – працюють за допомогою стандарту Bluetooth 4.0 Low Energy. Час роботи акумулятора конкретно взятого пристрою залежить від виробника.
- бікони, що живляться від USB порту – можуть отримувати енергію за допомогою стандартного USB-порту, що робить їх ідеальними для постійних установок.

Бікони досягають оптимального енергоспоживання, залишаючись в режимі сну більшість часу роботи, та прокидаючись лише у попередньо визначені інтервали часу. Інтервали можуть становити від 100 мс до декількох секунд і можуть регулюватися, залежно від призначення бікона. Наприклад, коротша трансляція сигналу, збільшує кількість пакетів, що транслюються, а це, в свою чергу покращує точність показань, але збільшує кількість споживаної енергії[5].

Дані, що транслюються BLE-біконом, містяться у вигляді пакетів відповідно до специфікації Bluetooth .

Протоколи BLE(рисунок 2.2) формують поля даних, розділяючи їх на невеликі сегменти, які містять інформацію про бікон, потужність сигналу, вихідну потужність та інше. Найпопулярнішими є протоколи iBeacon, AltBeacon та Eddystone. Вони мають різні структури пакетів, які подано на рисунку 3.5.

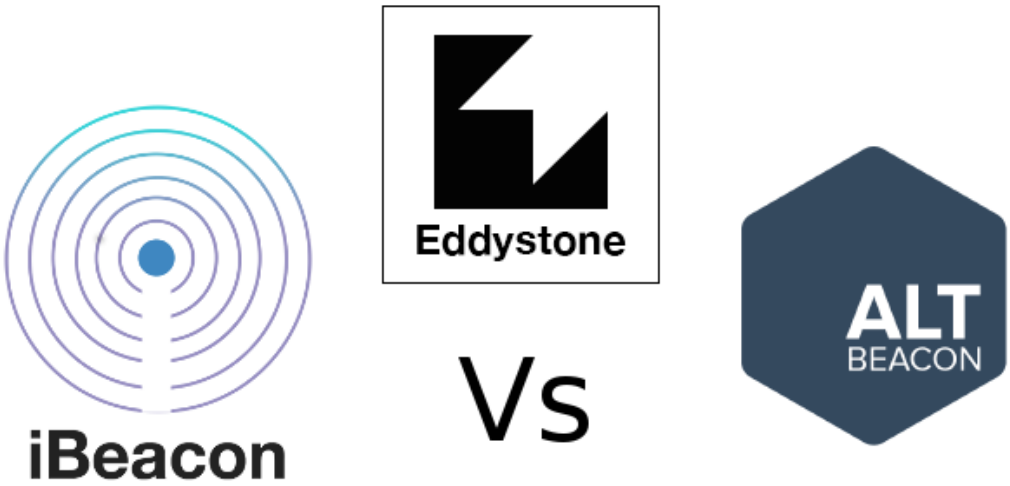

Рис. 2.2 – Протоколи BLE[7].

Протокол iBeacon, розроблений компанією Apple в 2013 році, був першим в технології. Компанія використовувала його, як спосіб передачі широкомовного сигналу між iOS пристроями . Ця система також використовувалася для оцінювання відстані між пристроями, що транслювали сигнал, до тих, що його приймали. Пізніше, були введені спеціальні бікони BLE, що усунуло необхідність мобільного пристрою. Проте, досі є можливість симулювати роботу бікону, для цього потрібно просто встановити додаток на пристрій, що має Bluetooth, та дати йому дозвіл на його використання[8].

Протокол AltBeacon визначає формат рекламного повідомлення, яке транслюють бікони. Специфікація AltBeacon безкоштовна для всіх, без винагороди та комісій[9].

Eddystone - це відкритий протокол бікона, розроблений Google, який має деякі особливості та численні переваги перед iBeacon, найважливішим з яких є його відкритий код та ліцензія. Поля даних Eddystone, показані на рисунку 2.3.

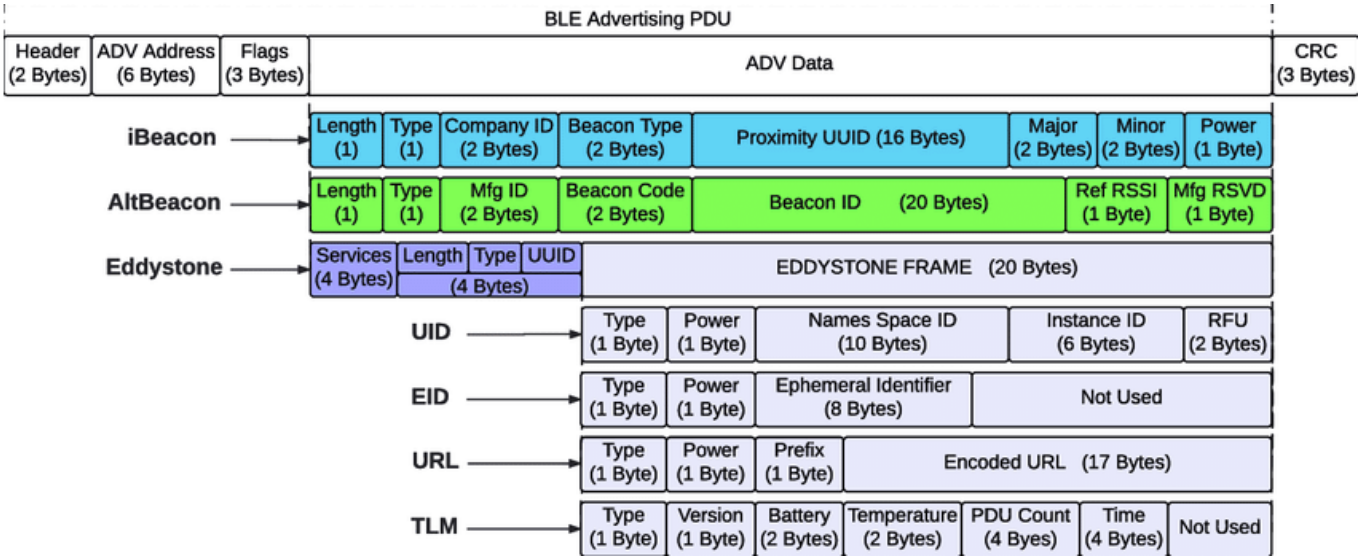

Рис. 2.3 – Структура пакетів iBeacon, AltBeacon та Eddystone[10].

Eddystone на відміну від інших стандартів визначає багато різних типів кадрів, а саме: UID, URL, TML, ETML, EID.

Бікони можуть використовувати ці формати одночасно разом або окремо. Вони містять таку ж інформацію, що і iBeacon, проте форматовані іншими способами. Більш детально структуру типів кадрів протоколу Eddystone, подано на рисунку 2.4.

Як згадувалося раніше, однією з переваг Eddystone перед iBeacon є ліцензування. Eddystone є крос-платформовим, його сигнал можуть ловити пристрої з різними операційними системами(рисунок 2.5), та вся інформація, необхідна для повного використання цього протоколу, надається компанією Google безкоштовно у своєму сховищі GitHub[11].

Рис. 2.4 – Типи Eddystone[7].

Рис. 2.5 – Крос-платформовість Eddystone[6].

Зазвичай пристрої Eddystone транслюються з інтервалом від 1 до 2 секунд, що особливо важливо, якщо бікон працює від акумулятора. Ще одним важливим моментом є те, що користувачам не потрібно встановлювати новий додаток для кожного місця, яке вони відвідують.

Кожен тип протоколу має свої особливості, проте є основні елементи, структура яких є спільною:

UUID - універсальний унікальний ідентифікатор. Він містить тридцять дві шістнадцяткових цифри, розділених на п’ять груп, що відокремлені дефісами, наприклад:

f7826da6-4fa2-4e98-8024-bc5b71e0893e

Кожна з 5 груп містить визначену кількість символі, а саме: перший розділ – 8, другий розділ – 4, третій розділ – 4, четвертий розділ – 4, п'ятий розділ – 12.

Major і Minor значення - це числа, що присвоюються бікону, для його ідентифікації з більшою точністю, у випадку коли створюється мережа біконів, з однаковим UUID.

Major і Minor мають значення цілочисельного типу без знаку в діапазоні від 0 до 65535. Для ідентифікації групи біконів використовують значення Major, а для конкретного пристрою в групі значення Minor.

2.3 Опис існуючих систем локалізації об’єктів

Міжнародний аеропорт Гонконгу є лідером в Азіатсько-Тихоокеанському регіоні з випробуваннями технології BLE та біконів, для надання інформації безпосередньо до мобільного пристрою пасажира. Систему впроваджено з метою вдосконалення та оптимізації маршрутів пересування пасажира аеропортом. Приклад роботи мобільного додатку подано на рисунку 2.6.

Технологія спрацьовує у відповідний час та у відповідному місці і відображає інформацію про поточне місцеположення на пристроях пасажирів.

За допомогою біконів аеропорт надає пасажирам всю необхідну для комфортних перельотів внутрішню інформацію, наприклад час проходу до входів та виходів, побудову оптимального маршруту до тієї чи іншої точки призначення в

середині аеропорту, доступ до зали очікування, а також попередження про наближення часу посадки на рейс.

Бікони допомагають зменшити кількість заторів, надаючи пасажирам точну та своєчасну інформацію. У свою чергу, це призводить до полегшення посадки для пасажирів, що дозволяє аеропорту збільшити кількість вильотів, а отже значно збільшити свій прибуток.

Постійне надання інформації про найкоротший шлях допомагає пасажирам дізнатися, скільки часу знадобиться, щоб дістатися до різних частин аеропорту, що дозволяє зменшити кількість запізнень на рейси[12].

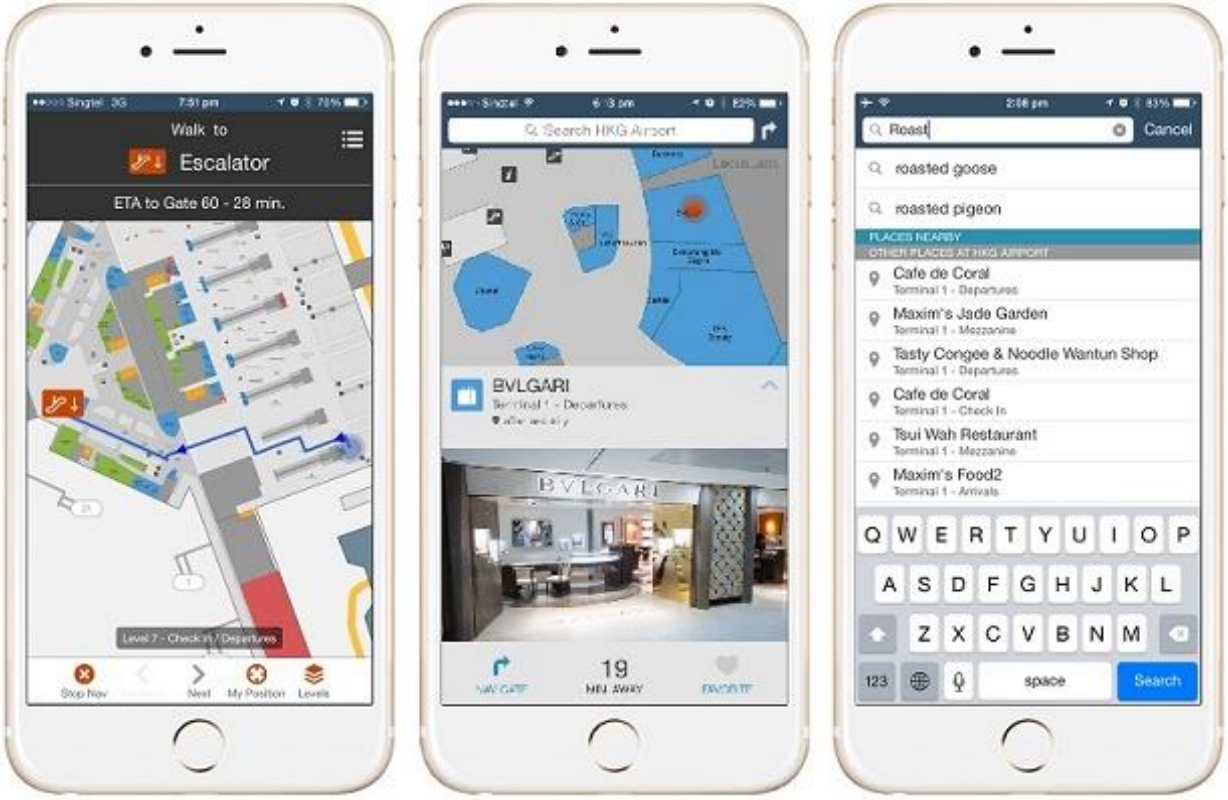

Рис. 2.6 – Мобільний додаток міжнародного аеропорту Гонконгу[12].

Один з магазинів відомої на весь світ мережі супермаркетів та гіпермаркетів «Ашан» розмістив близько двухсот біконів на своїй території. Бікони розсилають клієнтам інформацію про акції та знижки через встановлений мобільний додаток.

В момент, коли клієнт стоїть біля прилавків з одягом та аксесуарами він отримує повідомлення, від встановленого мобільного додатку, з інформацією про розпродаж одягу та вигідні пропозиції. Якщо клієнт мережі йде до зали з продуктовими товарами, то бікон сповістить його про акції даного відділу. Також покупці можуть переглянути місцезнаходження кожного з відділів і виконуючи вказівки додатку, можуть в найкоротший час знайти потрібні їм товари. Таким чином мережа підвищує свої прибутки та кількість клієнтів.

Одне з відділень компанії “Нова пошта”, яка надає різні послуги у сфері доставки документів, вантажів та відправлень в місті Києві тестує Bluetooth-технологію iBeacon. Клієнт, який заздалегідь встановив мобільний додаток “Нова пошта”, отримує повідомлення з детальною інформацією про його відправлення, як тільки заходить до відділення.

Відомий в усьому світі гіпермаркет «Carrefour» використовує BLE технологію з біконами у своїй мережі магазинів, яка розташована в Румунії.

Бікон-маркетинг надає клієнтам вичерпну інформацію про продукцію, акції і спеціальні пропозиції у різних відділах магазину цієї мережі. В результаті однієї з таких акцій, аналітика показала, шо їх офіційний мобільний додаток було встановлено на 400% частіше, а ніж раніше, а кількість користувачів зросла на понад 600% всього за 7 місяців.

Щороку безліч людей відвідує музеї та галереї по всьому світу. Відвідування музеїв може викликати багато проблем, через відсутність інформації про експонати на рідній мові. Особливо це стосується туристів, що мають поганий рівень англійської мови. Проте через погане інформування мало хто знає, що в багатьої відомих музеях та галереях світу вже давно впроваджені системи, що працюють за технологією BLE. Музеї та галереї розміщують бікони біля картин чи інших експонатів і його опис висилається відвідувачам на мобільний додаток на обраній ними мові.

3. ПРОГРАМНІ ЗАСОБИ РОЗРОБКИ СИСТЕМИ ЛОКАЛІЗАЦІЇ ОБ'ЄКТІВ

3.1 Вибір оптимальних технологій

За дослідженнями американської компанії, провайдера в сфері фінансової інформації Bloomberg, на восьми з десяти смартфонів встановлено операційну систему Android[13]. Таким чином, можна стверджувати, що вона є найбільш поширеною операційною системою для мобільних додатків. Отже, Android є оптимальним вибором для системи, яка буде впроваджується в навчальному закладі, тому що абсолютна більшість користувачів мобільних пристроїв використовують саме цю операційну систему[14].

Для розробки під Android існує декілька офіційних мов програмування:

1. Java – мова програмування, яка є офіційною для розробки для операційної системи Android, вона підтримується інтегрованим середовищем розробки AndroidStudio. Більшість додатків у Google Play написано саме на цій мові. Java код виконується віртуальною машиною, яка його інтерпретує у машинний.

2. Kotlin – мова програмування, що розроблена, як нова, більш проста для читання на базі Java. Вона так само інтерпретується віртуальною машиною Java, проте вирішує деякі проблеми, наприклад, одну з найбільш часто отримуваних помилок: виключення нульового показника англійською – Null Pointer Exception. Проте, основною перевагою мови Kotlin є значне зменшення шаблонного коду Java.

Таким чином, проаналізувавши інформацію, що подано вище, для розробки системи було обрано мову програмування Kotlin[15].

Як найбільш оптимальне середовище розробки для мобільних додатків було обрано IntelliJ IDEA Ultimate[16]. Серед інших представлених на ринку середовищ розробки воно має наступні переваги:

- розумне автодоповнення

- інструменти для аналізу якості коду
- зручна навігація
- підтримка всіх популярних фреймворків і платформ
- інтеграція з серверами додатків
- інструменти для роботи з різними базами даних, а також SQL файлами
- інструменти для тестування та аналізу коду, в тому числі підтримка всіх популярних фреймворків для тестування.
- зручний клієнт і редактор для схеми бази даних

Для створення бази даних обрано мову структурних записів SQL та систему керування базами даних MySQL[17], як безпрограшний варіант, що використовується для багатьох успішних проектів.

Для налагодження зв'язку між базою даних та мобільними додатками обрано скриптову мову програмування PHP та веб-додаток для адміністрування бази даних PHPMyAdmin[18]. Для редагування PHP запитів було обрано ще один з продуктів розробки компанії JetBrains, а саме PhpStorm[19] - інтегроване середовище розробки з інтелектуальним редактором, що підтримує всі версії PHP.

Для створення веб-додатку адмістратора обрано мову гіпертекстової розмітки HTML та спеціальну мову стилю сторінок CSS. У створенні веб-додатку, так само як і у створенні мобільних додатків, використовується скриптова мова програмування PHP.

3.2 Мова програмування Kotlin

Статично типізована мова програмування. Вона розробляється компанією JetBrains[20] та працює на базі JVM.

Мова може компілюватися в JavaScript, а це робить її підходящою не тільки для Android розробки, а й для розробки мобільних додатків для різних мобільних

платформ або навіть може бути скомпільована для роботи на таких операційних системах, як Windows, macOS та Linux.

Ця мова програмування розробляється з 2010 року, проте публічно вона була представлена лише в липні 2011 року.

З 17 травня 2017 року Kotlin входить до списку офіційно підтримуваних мов для розробки застосунків для платформи Android. З 7 травня 2019 року мова навіть обігнала свою попередницю Java, та стала пріорітетною для розробки Android застосунків, тому мобільні застосунки, написані на Java, почали втрачати свої позиції на ринку, в порівнянні з Kotlin додатками.

Синтаксис Kotlin використовує елементи не одніє тільки мови Java, тут також присутній синтаксис з таких широковідомих мов програмування, як Паскаль, TypeScript, Haxe, PL / SQL, F#, Go і Scala, C ++, Java, C#, Rust і D.

На відміну від Java, крапка з комою у Kotlin, як роздільник операторів є необов'язковою. При оголошенні змінних і параметрів наслідування від Java теж не выдбулося, типи даних вказуються після назви, а роздільником є двокрапка. Kotlin легко підтримує як процедурний стиль з використанням функцій, так і об'єктно-орієнтований підхід у програмуванні. Точкою входу до програми є функція main, яка зчитує масив параметрів з командного рядка.

Завдяки тому, що Kotlin повністю базується на Java, та працює на базі JVM, він може з легкістю компілюватись в Java і навпаки. В середовищі розробки Android Studio навіть є конвертер коду мови Java в Kotlin. Також є можливість використання коду обох мов один за одним, перетворення файлу Java в файл Kotlin. А під час роботи в новому файлі Kotlin є можливість вставити фрагменти коду Java і вони будуть автоматично перетворені[15].

3.3 Система автоматичного збирання Gradle.

Gradle — система автоматичного збирання, яка використовує предметно-орієнтовану мову на основі мови Groovy замість традиційної XML-подібної форми представлення конфігурації проекту.

Для визначення порядку виконання завдань проекту, Gradle використовує орієнтований ациклічний граф, в той час, як, наприклад, Apache Ant, ще одна з найбільш використовуваних систем автоматичного збирання визначає порядок виконання задач відносинами залежності.

Gradle призначено для побудови мультипроектів, які ростуть з часом, також він підтримує таку важливу річь, як інкрементальне збирання. Gradle легко визначає, які частини проекту зазнали змін, і виконує виключно ті задачі, які залежать від цих частин[21].

Далі, для прикладу, подано список основних Gradle залежностей які потрібні для більшості додатків:

```
dependencies {
   implementation "org.jetbrains.kotlin:kotlin-stdlib-jdk7:$kotlin_version"
   implementation 'androidx.appcompat:appcompat:1.1.0'
   implementation 'com.android.support:multidex:1.0.3'
   implementation 'androidx.core:core-ktx:1.2.0'
   implementation 'androidx.legacy:legacy-support-v4:1.0.0'
   implementation 'com.google.android.material:material:1.1.0'
}
```

3.4 Бібліотека Gson

Gson - це бібліотека Java/Kotlin з відкритим кодом для серіалізації та десеріалізації об'єктів Java/Kotlin до JSON. JSON це скорочення від JavaScript Object Notation, в перекладі з англійської — запис об'єктів JavaScript. JSON це текстовий формат обміну інформацією між системами. Він базується на звичайному

тексті та може бути легко прочитаним людиною. Таким чином, саме JSON є найбільш зручним форматом для передачі даних.

JSON надає змогу описувати структури даних, в тому числі об’єктів. Цей формат використовується переважно для передачі структурованої інформації через мережу[22].

3.5 Біліотека Android Jetpack Compose

Раніше весь інтерфейс користувача в Android базувався на класі View. В зв'язку з цим накопичилася безліч архітектурних недоліків, які потрібно було покращити та оновити. Але зробити це досить складно, не зламавши весь код, написаний на їх основі.

За останні роки з'явилося безліч нових концептів у світі клієнтських додатків, тому команда розробників компанії Google обрала радикальний шлях і повністю переписала рівень інтерфейсу користувача в Android з нуля. Так з'явилася відома всім Android розробникам бібліотека Android Jetpack Compose, що включає в себе концептуальні прийоми з React, Litho, Vue, Flutter та багатьох інших.

Для того щоб реалізувати інтерфейс користувача, доводиться значну частину часу приділяти роботі з різними типами файлів. Розмітку описувати у файлах з форматом xml, а потім використовувати Java / Kotlin код, щоб змусити її працювати. В інших xml-файлах задаються теми, анімація та навігація.

Використання Kotlin бібліотеки Android Jetpack Compose дозволяє писати інтерфейс користувача в декларативному стилі прямо в коді замість використання файлів з форматом xml. Проте, як показує практика, більшість розробників використовують обидва методи проектування інтерфейсу одочасно.

Xml файли зберігаються у спеціальній деректорії ресурсів. Деректорія ресурсів незалежно зберігає усі ресурси, що використовуються у мобільому додатку.

Документ з розширенням xml може керуватися класом Kotlin. Тобто отримавши доступ до такого файлу з середини класу, можна створювати та змінювати елементи інтерфейсу[23].

3.6 Середовище розробки IntelliJ IDEA

IntelliJ IDEA - в першу чергу IDE для таких мов, як Java та Kotlin, проте вона розуміє і надає інтелектуальну допомогу при написанні коду на SQL, JPQL, HTML, JavaScript і багатьох інших мовах.[16]

Середовище розробки доступне в двох варіантах: Community Edition і Ultimate Edition:

- Community Edition – повністю вільна і безкоштовна версія, в якій реалізовано повну підтримку Java SE, Kotlin, Android та інші. Також доступна інтеграція з найбільш популярними системами управління версіями. Така версія чудово підходить для студентів та початківців в Android розробці.
- Ultimate Edition – версія доступна лише під комерційною ліцензією, в якій реалізовано підтримку всього, що доступно в Community Edition, а також Java EE, UML-діаграм, підрахунок покриття коду, а також багато іншого[13].

3.7 Мова програмування SQL та СУБД MySQL.

SQL перекладається з англійської, як мова структурованих запитів. Використовується для формування запитів до реляційної БД, оновлення і керування нею. Створення схеми бази даних та її модифікації.

SQL не являє собою ані систему керування базами даних, ані окремий програмний продукт. На відміну від звичайних мов програмування, SQL може формувати інтерактивні запити, що є інструкціями для керування даними. Стандарт SQL містить функції для перевірки та захисту даних.

Існує багато систем керування базами даних, проте найбільш зручною з усіх є MySQL.

Система керування реляційними базами даних MySQL використовується, для створення динамічних веб-сторінок. Має чудову підтримку з боку різноманітних мов програмування[17].

3.8 Мова програмування PHP

З англійської PHP, тобто Hypertext Preprocessor перекладається, як гіпертекстовий препроцесор. Це мова програмування, яка використовується для генерації HTML-сторінок на стороні веб-сервера. PHP є однією з найпоширеніших мов, для використання у сфері веб-розробок. PHP підтримується переважною більшістю хостинг-провайдерів.

Веб-сервер інтерпретує PHP у HTML-код, який пересилається на сторону клієнта. На відміну від скриптової мови JavaScript, браузер отримує готовий html-код і клієнт не бачить PHP-код. З точки зору безпеки це є значною перевагою, тому що вірогідність несанкціонованого доступу до скриптів є мінімальною, проте це погіршує інтерактивність сторінок.

У PHP файлах часто містяться SQL запити до бази даних. PHP файл з запитом до бази даних має наступну структуру:

1. Файл починається з тегу відкриття:

```
<?php
```

2. Далі йдуть запити до інших PHP файлів, інформація з яких використовується у даному файлі:

```
require 'init.php';
```

3. Список параметрів:

```
$name = $_POST["name"];
$full_name = $_POST["full_name"];
$telephone = $_POST["telephone"];
$email = $_POST["email"];
```

```
$password = $_POST["password"];
$beacon_id = $_POST["beacon_id"];
```

4. SQL запит до бази даних:

```
$sql_query = "insert into teacher(name, full_name, telephone, e_mail, password,
beacon_id)
values('$name', '$full_name', '$telephone', '$email','$password', $beacon_id)";
```

5. Закриваючий тег:

```
?>
```

3.9 Середовище розробки PhpStorm

PhpStorm – це крос-платформове інтегроване середовище розробки для PHP. Розроблено його компанією JetBrains.

Редактор парцює з такими мовами, як PHP, JavaScript та HTML. Має можливості інтелектуального аналізу коду та автоматизовані засоби рефакторингу для PHP та JavaScript.

Також редактор містить повноцінний SQL-редактор, а отже надає можливість швидко, не переходячи до інших програмних продуктів, редагувати отримані результати запитів. Тому середовище розробки PhpStorm є одним з найбільш зручних середовищ для роботи з PHP-запитами до бази даних, з представлених на ринку. PhpStorm розроблений на основі платформи IntelliJ IDEA, яка написана на Java[19].

3.10 Система адміністрування PHP MyAdmin

Написаний на мові PHP, веб-додаток з відкритим кодом, представляє собою веб-інтерфейс для адміністрування СУБД MySQL.

PhpMyAdmin дозволяє через браузер і не тільки здійснювати адміністрування сервера MySQL, запускати команди SQL і переглядати вміст таблиць і баз даних, які

розташовано на віддаленому сервері. Додаток користується великою популярністю у веб-розробників, так як дозволяє управляти СУБД MySQL без безпосереднього введення SQL команд[18].

3.11 Мови HTML та CSS

HTML – стандартизована мова розмітки документів у Інтернеті. Вона інтерпретується браузерами, а форматований і отриманий в результаті інтерпретації текст відображається на екрані монітора комп'ютера або мобільного пристрою.

HTML-сторінки, найчастіше, передаються браузеру від сервера по двох основних протоколах HTTP або HTTPS, у вигляді простого тексту, у випадку HTTP або з використанням шифрування, у випадку HTTPS.

CSS – формальна мова для опису зовнішнього вигляду документа, який було написаного з використанням мови розмітки. Переважно використовується як засіб опису, оформлення зовнішнього вигляду веб-сторінок, написаних за допомогою мови розмітки HTML.

У файлах CSS стилів зберігаються дані про стиль HTML-елементів. Приклад можливого стилю для HTML-елементу <form> подано нижче:

```
form {
   width: 90%;
   margin: 10px auto;
   text-align: left;
   padding: 10px;
   border: 1px solid #026B5A;
   border-radius: 5px;
}
```

Таким чином файли стилів допомагають запобігти дублювання коду, котрий потрібен для створення привабливих інтерфейсів, тож за їх допомогою, можна легко використовувати один й той самий стиль на різних HTML сторінках веб-додатків.

4. ОПИС ПРОГРАМНОЇ РЕАЛІЗАЦІЇ

4.1 Алгоритм роботи системи

Система складається з двох мобільних додатків, бази даних та веб-додатку адміністратора.

Можливості кожної системи було визначено шляхом створення діаграми прецедентів розроблюваної системи. Діаграма прецедентів або по іншому діаграма варіантів використання показує можливі варіанти використання системи користувачами.

Діаграма варіантів використання або діаграма прецедентів – показує взаємодію між користувачами та програмною системою. Користувачів прийнято називати акторами, а їх можливі дії з системою – варіантами використання.

Акторами можуть бути і люди, і різноманітні зовнішні системи та пристрої. Для розробленої системи акторами є: викладач, студент та адміністратор бази даних.

Кожен актор має певні варіанти використання системи.

Викладач має можливість:

- Переглянути розклад занять на сьогодні
- Переглянути успішність студента
- Переглянути успішність групи
- Переглянути мапу навчального закладу
- Переглянути інформацію про студента

Студент у мобільному додатку може виконати наступні дії:

- Переглянути розклад занять на сьогодні
- Переглянути інформацію про викладача
- Переглянути розклад викладача на сьогодні
- Переглянути мапу навчального закладу
- Переглянути інформацію про студента

Важливим уточненням є те, що студенту не потрібно виконувати надлишкових дій, для того, щоб поставити відмітку про свою присутність. Завдяки технології BLE, мобільний додаток сам виконає всі потрібні дії. Для отримання відмітки про присутність, достатньо підійти до викладача, у котрого є бікон, під час заняття.

Адміністратор бази даних виконує адміністрування бази даних, за допомогою веб-додатку з відкритим кодом для адміністрування баз даних - PHPMyAdmin.

Діаграму прецедентів подано на рисунку 4.1.

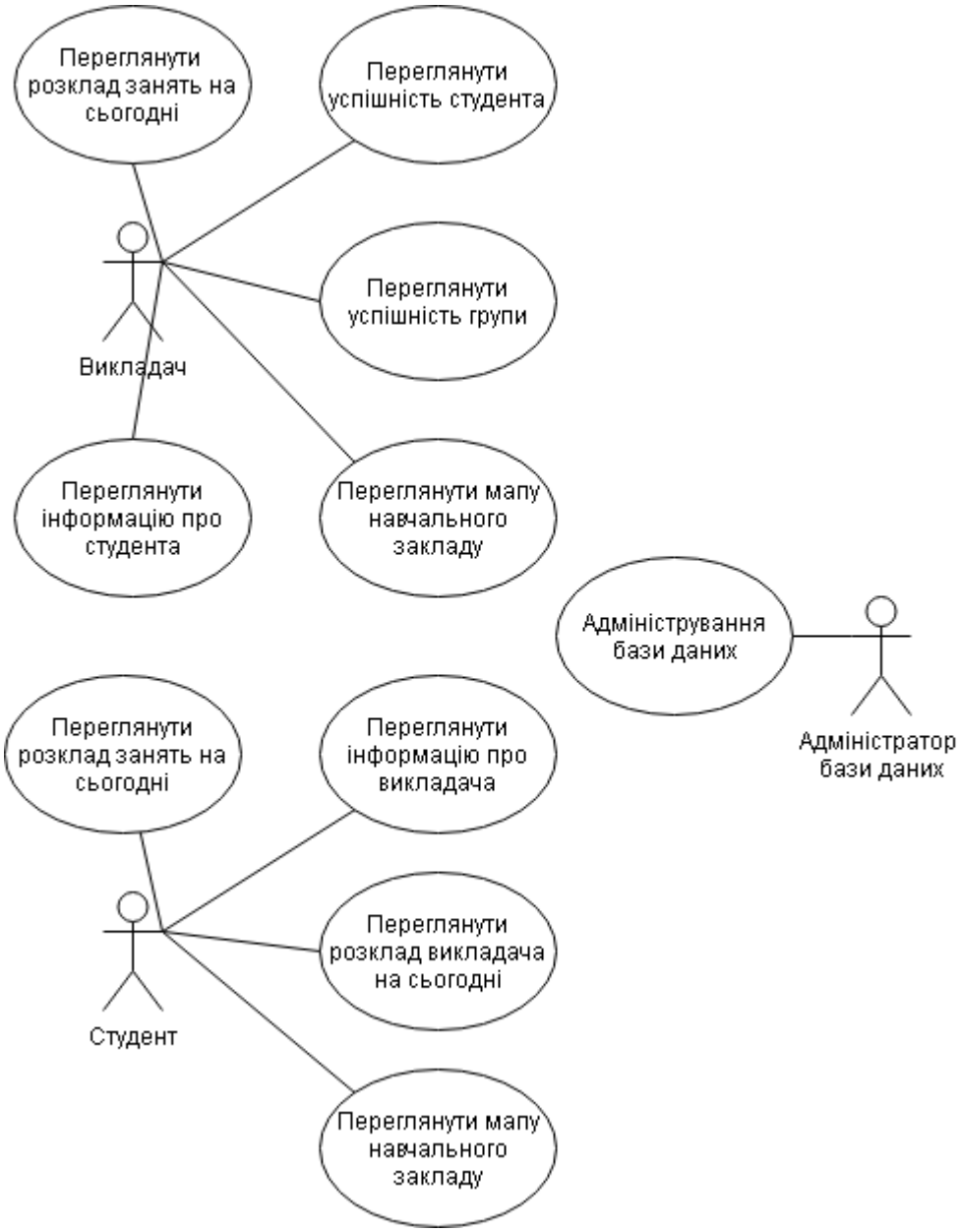

Рис 4.1 – Діаграма варіантів використання.

1. Кожен викладач навчального закладу, у якому впроваджена система, має особистий бікон з унікальним ідентифікатором, який транслює сигнал для студентів, що присутні в аудиторії.

2. Кожен студент має змогу ввімкнути мобільний пристрій, на якому встановлено додаток для відслідковування відвідування, зайти до нього та авторизуватися. Після чого він отримує доступ до поточного розкладу та може переглянути список поточних занять.

3. Додаток студента фіксує сигнал, який транслюється біконом викладача й у певний час, за розкладом, студент має можливість поставити відмітку про свою присутність в аудиторії, для цього, йому потрібно лише ввімкнути мобільний додаток, який автоматично відправить інформацію про його присутність до бази даних.

4. Викладач має можливість переглянути відмітки про присутність студентів.

5. Редагуванням бази даних відвідування навчального закладу, реєстрація нових викладачів, студентів, предметів та біконів, які використовує кожен викладач, здійснюється адміністратором бази даних, за допомогою веб додатку для адміністрування.

Функціонал програмних підмодулів:

Мобільний додаток студента – надає змогу студенту переглянути список занять на сьогодні, інформацію про викладачів та їх поточний розклад, поставити відмітку про присутність на занятті, а також переглянути мапу навчального закладу, для зручного пересування студмістечком.

Мобільний додаток викладача – надає змогу викладачеві переглянути список занять на сьогодні, інформацію про студентів, що навчаються на його курсах та відвідування ними занять, як індивідуально для кожного студента, так і відмітки студентів по групах.

База даних – зберігає інформацію про студентів, викладачів, розклад занять та відвідування студентами занять.

Веб-додаток – призначений для адміністрування бази даних системи. А саме внесення нових записів, редагування та видалення вже існуючих записів у базі даних.

Редагуванням бази даних займається адміністратор бази даних.

На рисунку 4.2 подано діаграму розгортання розробленої системи. На діаграмі відображаються обчислювальні вузли програмної системи під час використання. А також компоненти, та об'єкти, які безпосередньо виконуються на цих вузлах.

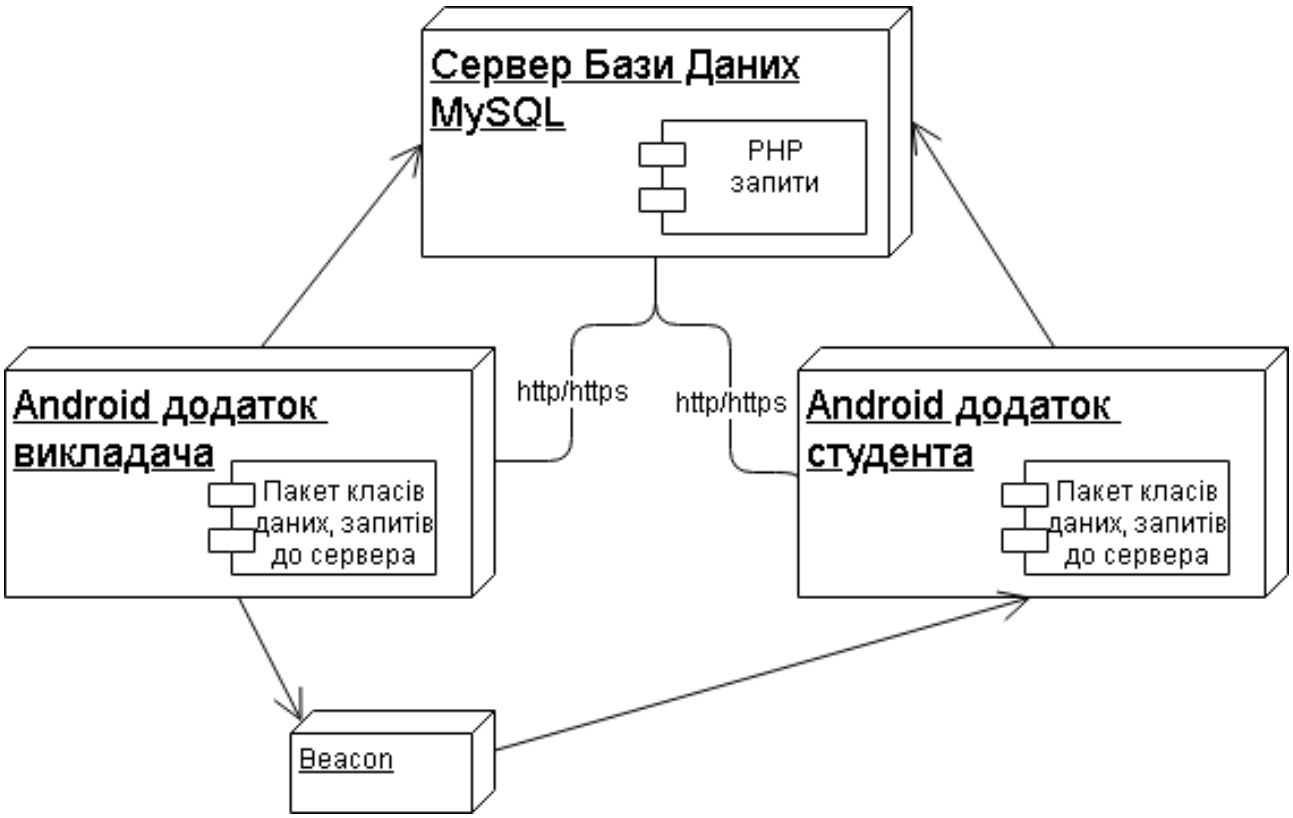

Рис. 4.2 – Діаграма розгортання системи

Першим кроком до створення повноцінної системи обліку відвідування стало проектування потрібної бази даних, адже для початку розробки мобільних додатків студента та викладача, вона є необхідною в першу чергу.

Також першочерговою задачею є створення зручного додатку для адміністрування бази даних.

4.2 Опис бази даних

Для успішної роботи системи абсолютно необхідною є база даних, котра містить інформацію про деякі аспекти навчального процесу у навчальному закладі, де вона впроваджується. У нашому випадку ключовими особами для проведення обліку відвідування є викладачі та студенти. Зважаючи на використання технології BLE, котра надасть можливість легко локалізувати місцезнаходження викладача необхідним є збереження інформації про бікони – пристрої, які будуть надані кожному викладачу, який є користувачем системи. Ще одним важливим аспектом є розклад, тож його зберігання є ще однією необхідною інформацією.

Отже, було прийнято рішення спроектувати базу даних, котра б містила інформацію про:

- Розклад занять навчального процесу
- Предмети, що викладаються у навчальному закладі
- Викладачів, які працюють в навчальному закладі
- Студентів, проходять навчання у навчальному закладі
- Розподіл студентів за групами та факультетами
- Бікони, які використовуються викладачами, для обліку відвідування

Базу даних створено засобами мови програмування SQL та за допомогою системи управління базами даних MySQL.

База даних складається з 11 таблиць:

Таблиця “Заняття” містить інформацію про заняття (таблиця 4.1.).

Таблиця 4.1. Структура таблиці “Заняття”

Назва поля	Тип поля	Призначення
ідентифікатор	INT	автоінкремент, унікальне поле в

		таблиці
ідентифікатор предмета у викладача	INT	ідентифікує предмет з таблиці “Предмети”
ідентифікатор часу заняття	INT	ідентифікує час заняття з таблиці “Час заняття”
ідентифікатор дня заняття	INT	ідентифікує день заняття з таблиці “День заняття”

Таблиця “Бікон” – містить дані про бікони, що використовуються у навчальному закладі, для забезпечення локалізації(таблиця 4.2.).

Таблиця 4.2. Структура таблиці “Бікон”

Назва поля	Тип поля	Призначення
ідентифікатор	INT	автоінкремент, унікальне поле в таблиці
унікальний ідентифікатор групи біконів	WARCHAR	унікальний ідентифікатор, за яким можна однозначно визначити бікон
мінор	INT	мінорне значення бікона
мажор	INT	мажорне значення бікона

Таблиця “Студент” місить інформацію про студентів, які навчаються у навчальному закладі(таблиця 4.3.).

Таблиця 4.3. Структура таблиці “Студент”

Назва поля	Тип поля	Призначення
ідентифікатор	INT	автоінкремент, унікальне поле в таблиці
ім’я	WARCHAR	ім’я студента
ідентифікатор групи	INT	ідентифікує групу з таблиці “Група”
телефон	WARCHAR	телефон студента
електронна пошта	WARCHAR	електронна пошта студента

пароль	WARCHAR	пароль студента

Таблиця “Викладач” містить інформацію про викладачів, які працюють у навчальному закладі (таблиця 4.4.).

Таблиця 4.4. Структура таблиці “ Викладач ”

Назва поля	Тип поля	Призначення
ідентифікатор	INT	автоінкремент, унікальне поле в таблиці
ім’я	WARCHAR	ім’я викладача
повне ім’я	WARCHAR	повне ім’я викладача
телефон	WARCHAR	телефон викладача
електронна пошта	WARCHAR	електронна прошта викладача
пароль	WARCHAR	пароль викладача
ідентифікатор бікону	INT	ідентифікує бікон з таблиці “Бікон”

Таблиця “Група” містить дані про навчальну групу у навчальному закладі(таблиця 4.5.).

Таблиця 4.5. Структура таблиці “Група”

Назва поля	Тип поля	Призначення
ідентифікатор	INT	автоінкремент, унікальне поле в таблиці
код	WARCHAR	код групи
факультет	WARCHAR	факультет
курс	INT	курс
освітній ступінь	WARCHAR	освітній ступінь, до якого відноситься група

Таблиця “ Предмет ” містить інформацію про предмети, які викладаються у навчальному закладі(таблиця 4.6.).

Таблиця 4.6. Структура таблиці “Предмет”

Назва поля	Тип поля	Призначення
ідентифікатор	INT	автоінкремент, унікальне поле в таблиці
назва	WARCHAR	назва предмету

Таблиця “Час заняття” містить дані про час заняття за розкладом, для зменшення зберігання надлишкової інформації (таблиця 4.7.).

Таблиця 4.7. Структура таблиці “ Час заняття ”

Назва поля	Тип поля	Призначення
ідентифікатор	INT	автоінкремент, унікальне поле в таблиці
початок заняття	DATE	дата початку заняття
кінець заняття	DATE	дата кінця заняття

Таблиця “ День заняття” містить дані дні заняття, для зменшення кількості надлишкової інформації в базі даних(таблиця 4.8.).

Таблиця 4.8. Структура таблиці “День заняття”

Назва поля	Тип поля	Призначення
ідентифікатор	INT	автоінкремент, унікальне поле в таблиці
назва	WARCHAR	назва дня заняття

Таблиця “Предмети у викладачів” (таблиця 4.9.) зв’язує предмети та викладачів.

Таблиця 4.9. Структура таблиці “Предмети у викладачів”

Назва поля	Тип поля	Призначення
ідентифікатор	INT	автоінкремент, унікальне поле в таблиці
ідентифікатор предмета	INT	ідентифікує предмет з таблиці

		“Предмети”
ідентифікатор викладача	INT	ідентифікує викладача з таблиці “Викладач”
ідентифікатор групи	INT	ідентифікує групу з таблиці “Група”
тип заняття	WARCHAR	тип заняття
місце заняття	WARCHAR	місце, де проходить заняття

Таблиця “Відвідування” (таблиця 4.10.) містить дані про відвідування студентами занять.

Таблиця 4.10. Структура таблиці “ Відвідування ”

Назва поля	Тип поля	Призначення
ідентифікатор заняття	INT	автоінкремент, унікальне поле в таблиці
ідентифікатор студента	INT	ідентифікує студента з таблиці “Студент”
дата заняття	DATE	дата дня заняття

Таблиця “Адміністратор” (таблиця 4.11.) містить дані про адміністраторів бази даних.

Таблиця 4.10. Структура таблиці “ Адміністратор”

Назва поля	Тип поля	Призначення
ідентифікатор	INT	автоінкремент, унікальне поле в таблиці
логін	WARCHAR	ідентифікує логін адміністратора
пароль	WARCHAR	Пароль адміністратора

Після створення таблиць бази даних, було налагоджено логічні зв’язки між ними. На рисунку 4.3 подано схему бази даних зі зв’язками.

Після створення бази даних локально, було прийнято рішення знайти хостинг для подальшої роботи та впровадження системи в реальних умовах. Після знаходження підходящого хостинг сервісу, базу даних було експортовано з

локального доступу та перенесено на сервер. Для подальшого налагодження роботи з базою даних використовувався веб-додаток для адміністрування баз даних – PHPMyAdmin.

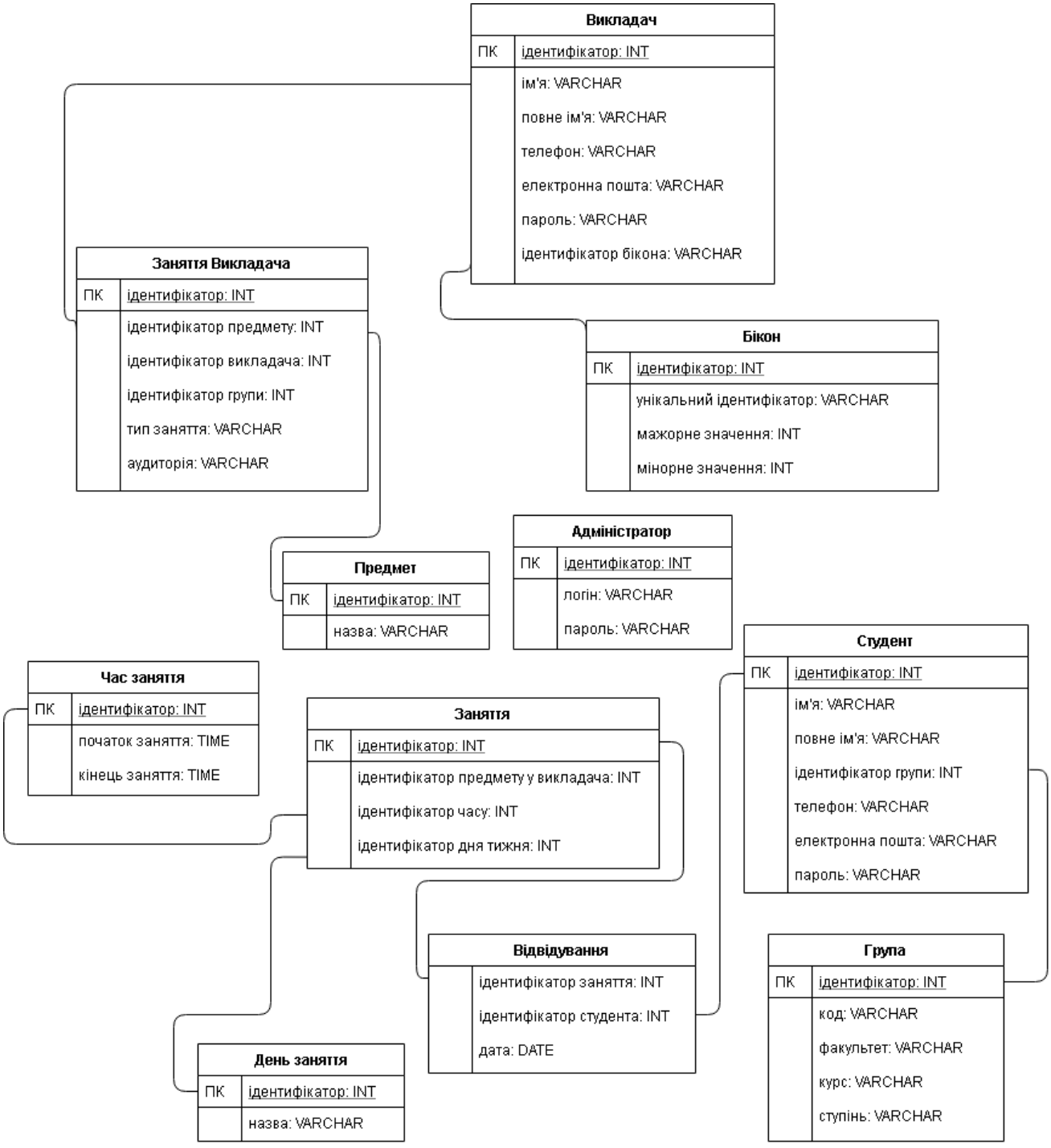

Рис. 4.3 - Схема бази даних

4.4 Веб-додаток адміністратора

Для зручного адміністрування бази даних було створено веб-додаток для адміністрування. Таким чином, адміністратор системи може додавати, редагувати, а також видаляти записи і для цього йому не потрібно знати мов програмування та робити це напряму на сервері, шляхом SQL запитів.

Для створення веб-додатку адміністратора було використано мову гіпертекстової розмітки HTML, спеціальна мова стилю сторінок CSS, а також скриптову мову програмування PHP.

Отже PHP файли зберігаються на сервері й під час запиту з браузеру за веб-адресою відкривається веб-сайт, який складається зі згенерованих з PHP скриптів, HTML сторінок.

Структура веб-додатку по суті являє собою сайт, за допомогою якого можна проводити адміністрування.

4.5 Мобільні додатки студента та викладача.

Мобільні додатки студента та викладача реалізовано засобами мови програмування Kotlin, у інтегрованому середовищі розробки IntelliJ IDEA Ultimate. Та з використанням універсального засобу розробки мобільних додатків для операційної системи Android - Android SDK.

Першим питанням у створенні мобільних додатків стало налагодження зв’язку зі створеною базою даних.

Зв'язок з базою даних

Для налагодження зв’язку між базою даних та мобільним додатком, таблиці з бази даних під час роботи мобільного додатку, записуються до об’єктів спеціальних класів даних (Data Class). Клас даних відрізняється від звичайного тим, що в ньому автоматично створюються такі функції як: toString(), equals(), hashCode(), copy().

Такі класи найбільш часто використовуються для подальшої роботи з форматом даних JSON. Саме для цієї цілі вони створені у додатках студента та викладача.

Для створення об’єкту класу даних, дані завантажуються за допомогою спеціального класу AsyncTask, в тілі якого виконується php-запит до бази даних, яка зберігається на сервері.

AsynkTask – це простий і зручний механізм, що переміщує трудомістку задачу виконання php-запиту в фоновий потік. Завдяки такому механізму вдалося легко синхронізувати обробники подій з графічним потоком.

Таким чином, оновлення елементів інтерфейсу користувача виконується лише, коли задача AsynkTask завершена.

Наприклад, для отримання дата класу Teacher, який містить інформацію про викладачів, створюється новий екземпляр класу LoadTeacher, в функції doInBackground() якого, виконується php-запит до бази даних і його результат записується до екземпляру класу даних Teacher:

```
val loadTeacher = LoadTeacher(this)
val loadTeacherExecute = loadTeacher.execute(message)
val teacher: Teacher? = loadTeacherExecute.get()
```

Клас LoadTeacher є асинхронною задачею та наслідує клас AsynkTask. Php-запити у відповідь повертають або відповідь у String форматі, або у форматі JSON, залежно від інформації, що запитувалася.

Там чином, додаток у фоновому режимі отримує потрібну інформацію з бази даних, за наступним алгоритмом:

1. У класі, який є ассинхронною задачею створюється функція doInBackground().
2. Створюється змінна, що зберігає інформацію про адресу сервера.
3. Створюється HTTP з’єднання з базою даних та відбувається його налаштування.
4. Створюється потік запису, до якого, за наявності, передаються параметри запиту.

5. Запит з відомими параметрами відправляється на сервер.

6. Створюється потік зчитування, котрий залежно від типу запиту, зчитує або змінну типу String, або дані у форматі JSON.

7. Функція doInBackground() повертає дані виклику.

Таким чином вдалося легко реалізувати зв'язок мобільних додатків з віддаленим сервером бази даних, на якому зберігаються попередньо написані PHP скрипти з запитами до БД.

Наступним завданням стало створення основного двіючого модулю додатку, в якому б відбувались всі взаємодії між інтерфесом користувача та блоком звязку з базою даних.

Основний модуль

Для створення мобільного додатку в Kotlin основну роль грають так звані Activity(Активності). Activity – це ключові файли в додатку, тому що саме тут відбуваються всі взаємодії з інтерфейсами, блоком взаємодії з базою даних та іншими активностями.

Для додатку студента було створено наступні активності:

LoginActivity – точка входу до мобільного додатку. Саме тут відбувається авторизація студента.

NavigationDrawerActivity – для реалізації бокового меню, а отже й логіки навігації між активностями.

ScheduleActivity – відображається розклад студента на поточний день тижня.

TeachersActivity – відображається список викладачів, у яких студент проходить певні курси.

TeacherInfoActivity – відображається інформація про окремо взятого викладача, а також його розклад на поточний день.

MapActivity – відображається мапа студмістечка.

Для додатку викладача створено наступні активності:

LoginActivity – авторизація викладача.

NavigationDrawerActivity – для навігації між активностями.

ScheduleActivity – відображається розклад викладача на поточний день тижня.

StudentActivity – відображається список студентів, по групах, у яких викладач проводить певні курси.

StudentInfoActivity – відображається інформація про окремо взятого студента, а також результати його відвідування

SubjectActivity – відображається список предметів та груп, які проходять курси кожного з предметів викладача

SubjectInfoActivity – відображається список відвідування конкретної групи студентів

MapActivity – відображається мапа студмістечка

Отже весь додаток, по суті, складається зі змінюючих одна одну активностей. Також саме у цих активностях відбуваються всі виклики асинхронних задач для зв'язку з базою даних. І саме завдяки механізму Activities в Android, інформація з бази даних легко впроваджується і інтерфейс користувача та ,навпаки, дані отримані від користувача – легко записати до бази даних.

Кожна активність тісно пов'язана з XML файлом, який визначає розмітку та дизайн екрану, для цієї активності.

Інтерфейс користувача

Базовий інтерфейс користувача в Android створено за допомогою спеціальних XML файлів. Тобто файлів написаних на розширюваній мові розмітки(англ. Extensible Markup Language).

Кожен такий файл починається з декларації, у якій вказується версія XML та кодування документу.

Кожен файл обов'язково має кореневий елемент. Для більшості XML файлів у мобільних додатках студента та викладача обрано <LinearLayout>, тому що саме з нього можна побудувати більшість потрібних дизайнів для активностей.

Кожен кореневий елемент, у загальному випадку має ряд дочірніх елементів, які більш детально описують структуру екрану.

Для доступу до ресурсу, а XML файли зберігаються саме в директорії ресурсів, в активності, яка обслуговує даний XML файл, необхідно створити посилання на ідентифікатор ресурсу. Для завдання всіх ідентифікаторів у проекті використовується клас R, який створюється автоматично.

Також для цих цілей використано бібліотеку Android Jetpack Compose, яка дозволила створювати елементи інтерфейсу користувача в декларативному стилі прямо в коді, під час виконання програми.

Саме завдяки цій бібліотеці є можливим оновлення інформації на інтерфейсі користувача, безпосередньо під час роботи додатку.

Робота з біконами

Для роботи з біконами було використано бібліотеку - Android Beacon Library.

Вона надзвичайно легко впроваджується у додаток, а для її підключення, у випадку коли ви використовуєте систему автоматичного збирання Gradle, достатньо просто додати посилання на неї у списку залежностей:

implementation 'org.altbeacon:android-beacon-library:2+'

Бібліотека легко настроюється для роботи з iBeacon, Eddystone та іншими форматими біконів.За замовчуванням бібліотека налаштована на роботу з відкритим форматом AltBeacon.

Роботу з BLE біконами реалізовано за допомогою класу BeaconParser. При створені нового об’єкту класу BeaconParser, він автоматично починає відслідковувати бікони, котрі потрапили у діапазон зчитування. Для успішної роботи з біконами, при встановленні додатку обов’язково запитується дозвіл на відслідковування локації пристрою. У випадку, коли дозвіл дано не було, додаток не зможе корректно працювати.

Бібліотека також надала можливість налаштувати оптимальний інтервал сну, коли додаток не відслідковує присутні поруч бікони, а отже зменшує затрати енергії, які потребує додаток.

Отже, бібліотека дала можливість легко перетворити мобільний додаток студента на сканер BLE сигналу.

Детальний лістинг роботи з біконами додатку студента, наведено у Додатку 2, до цієї дипломної роботи.

На рисунку 4.3, подано діаграму компонентів додатку викладача. Діаграма відображає залежності між компонентами програмного продукту.

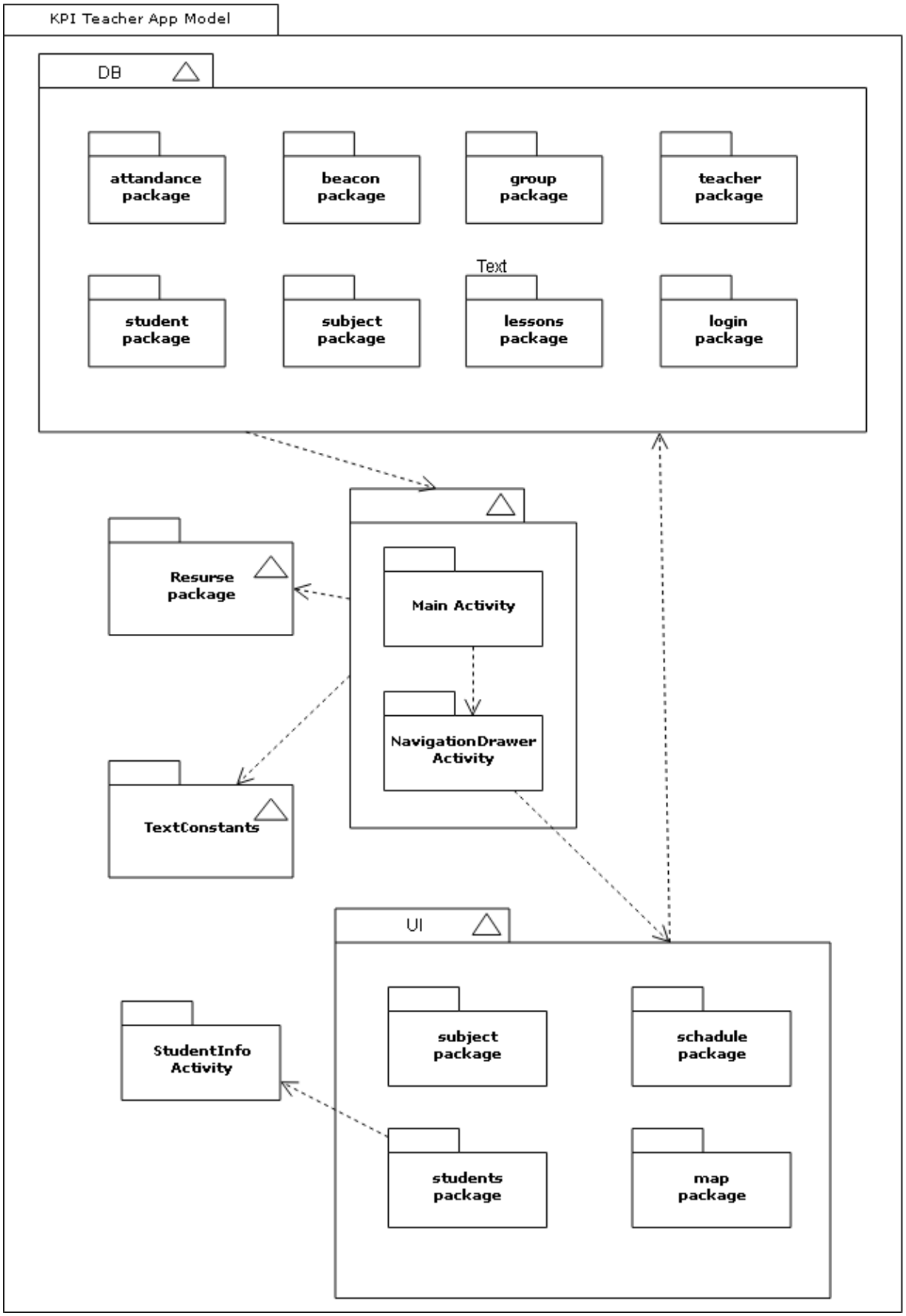

Рис. 4.3 – Діаграма компонентів додатку викладача

На рисунку 4.4, подано діаграму компонентів додатку студента:

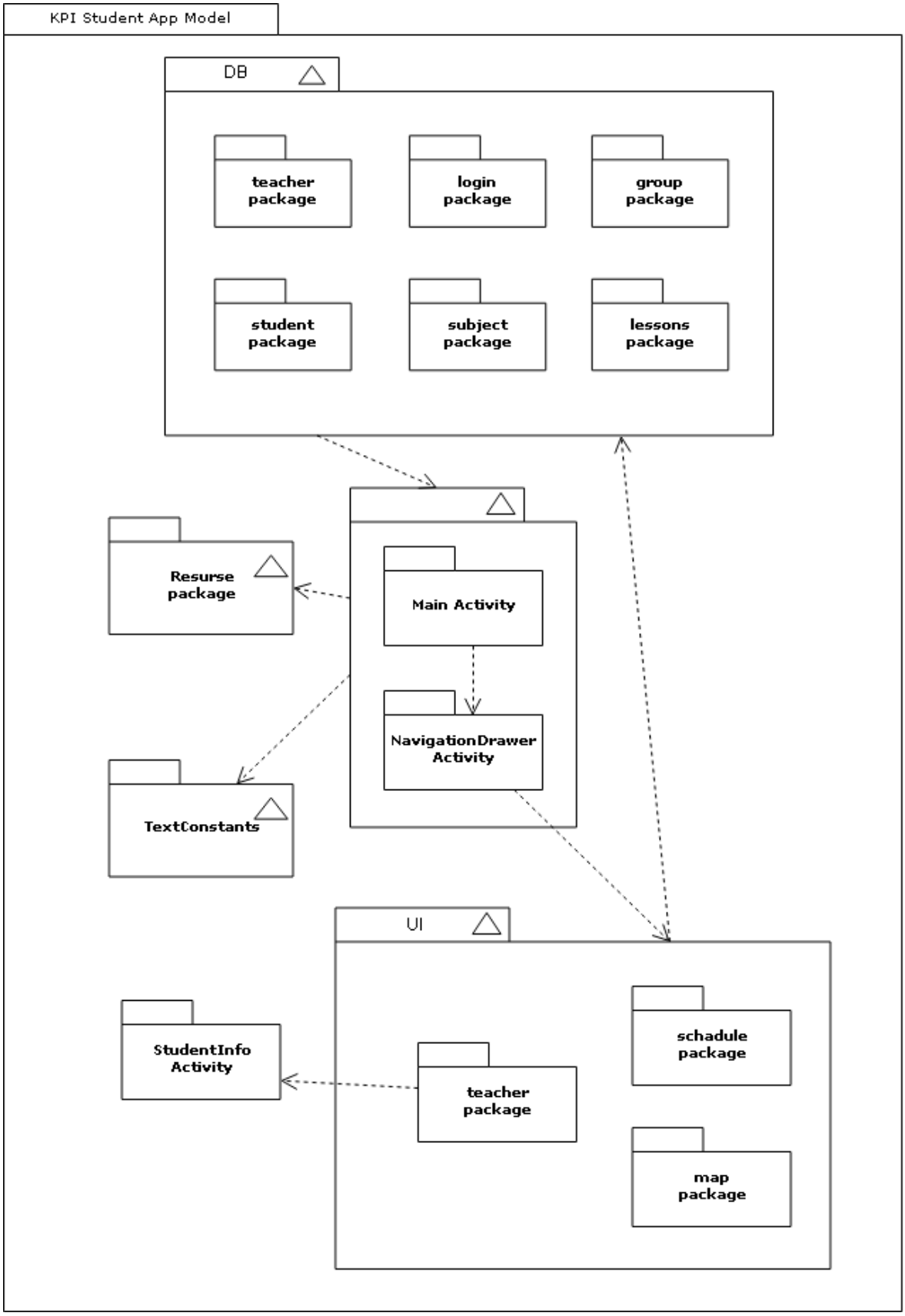

Рис. 4.4 – Діаграма компонентів додатку студента

5. РОБОТА КОРИСТУВАЧА З ПРОГРАМНОЮ СИСТЕМОЮ

5.1 Мобільний додаток студента

При відкритті мобільного додатку, першим, що бачить користувач є форма авторизації. Її подано на рисунку 5.1.

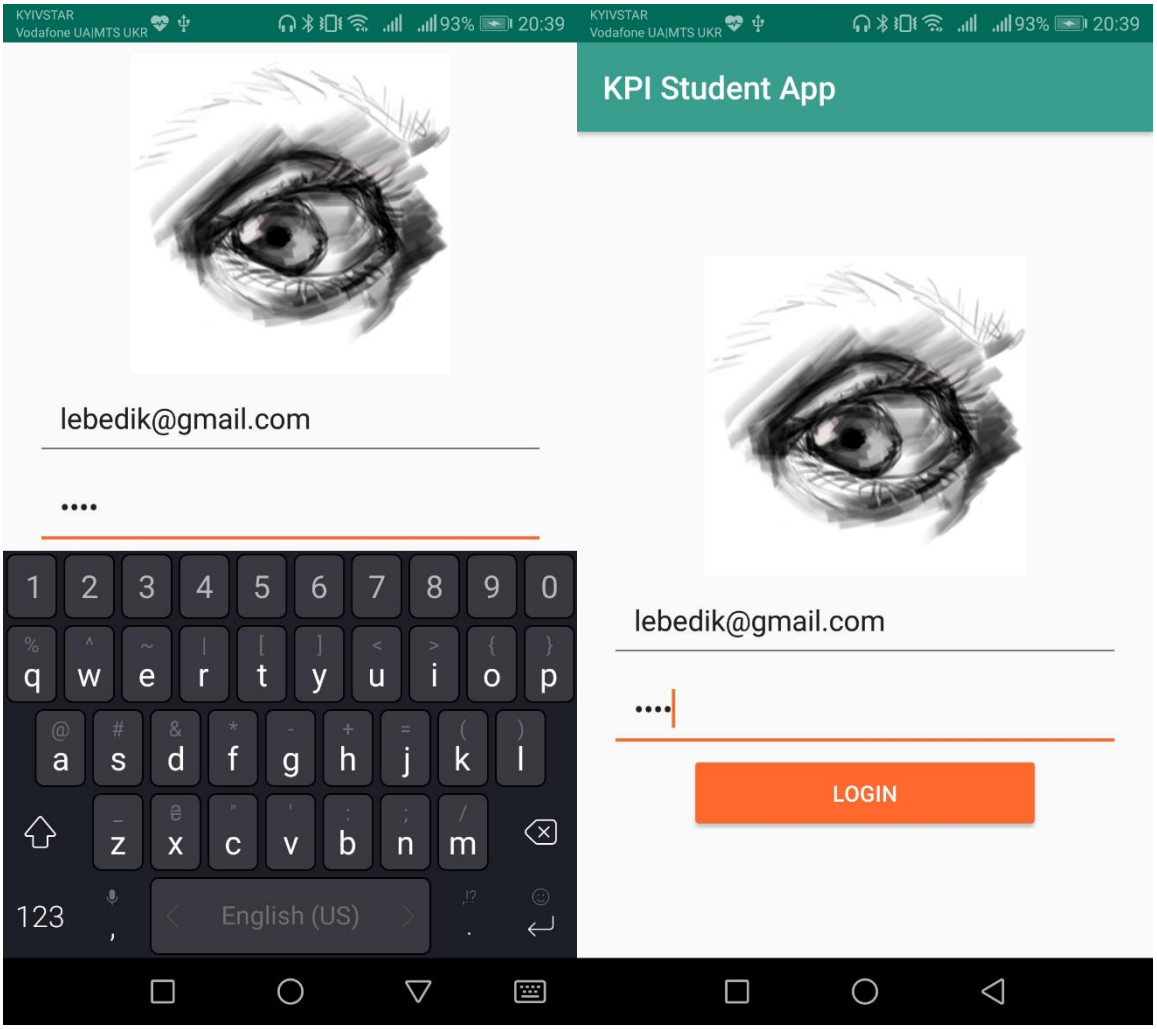

Рис 5.1 - Форма авторизації студента

Після авторизації, користувач системи потрапляє на сторінку з розкладом. Її подано на рисунку 5.2. Для того, щоб поставити відмітку про присутність на занятті, потрібно дочекатися часу початку лекції, підійти до викладача, котрий має бікон.

При натисканні оранжевої кнопки, що розташована з правого нижнього краю екрану, ініціюється перевірка на можливість поставити відмітку про відвідування поточного заняття. Якщо поряд немає бікону, потрібного викладача, або час заняття закінчився, ви отримаєте повідомлення: “ Немає доступних для відмітки занять”, в іншому випадку: “ Присутність підтверджено”.

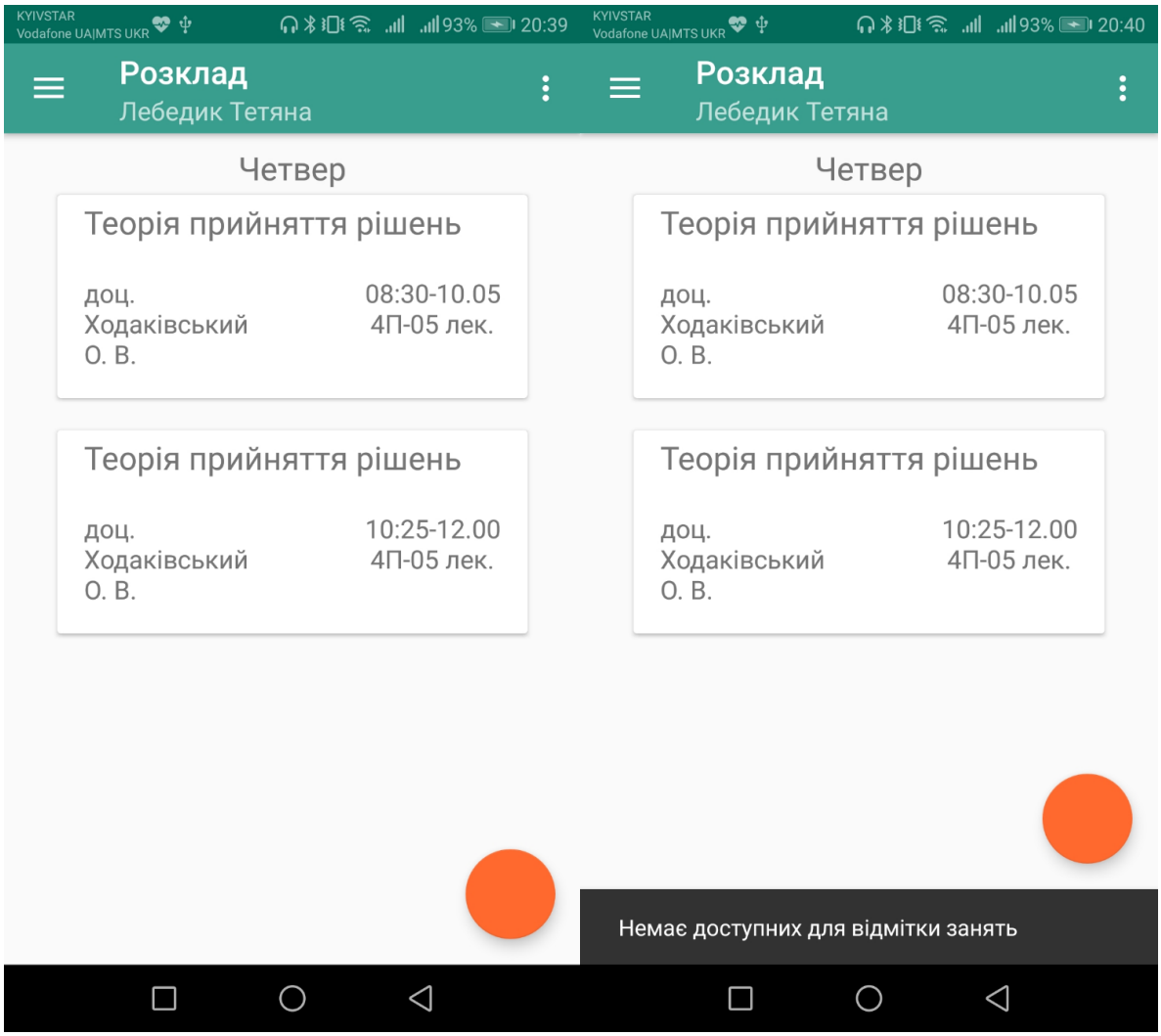

Рис 5.2 - Розклад студента на поточний день

За допомогою бокового меню, можна перейти на інші вкладки мобільного додатку. Бокове меню представлено на рисунку 5.3.

Бокове меню має наступні вкладки: “Розклад”, “Викладачі” та “Мапа”.

Мапу Київського Політехнічного Інституту було додано, для зручності пересування студмістечком та полегшенню орієнтування кампусом. Вкладку “Мапа” подано на рисунку 5.4.

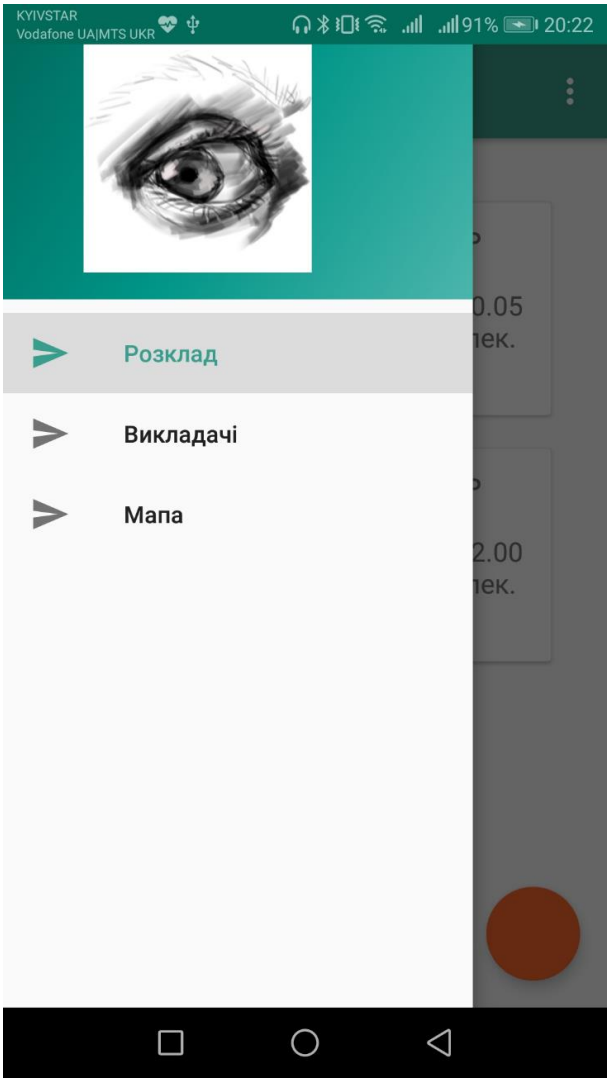

Рис 5.3 - Бокове меню додатку

Рис 5.4 - Мапа КПІ

Вкладка “Викладачі” призначена для перегляду переліку викладачів студентами. ЇЇ подано на рисунку 5.5.

Натиснувши на ім’я викладача, користувач має можливість переглянути більш детальну інформацію про нього: ім’я, прізвище та по батькові повністю, його телефон та електронну адресу. Також, корисною є інформація про поточні заняття викладача. Приклад детальної інформації про викладача подано на рисунку 5.6.

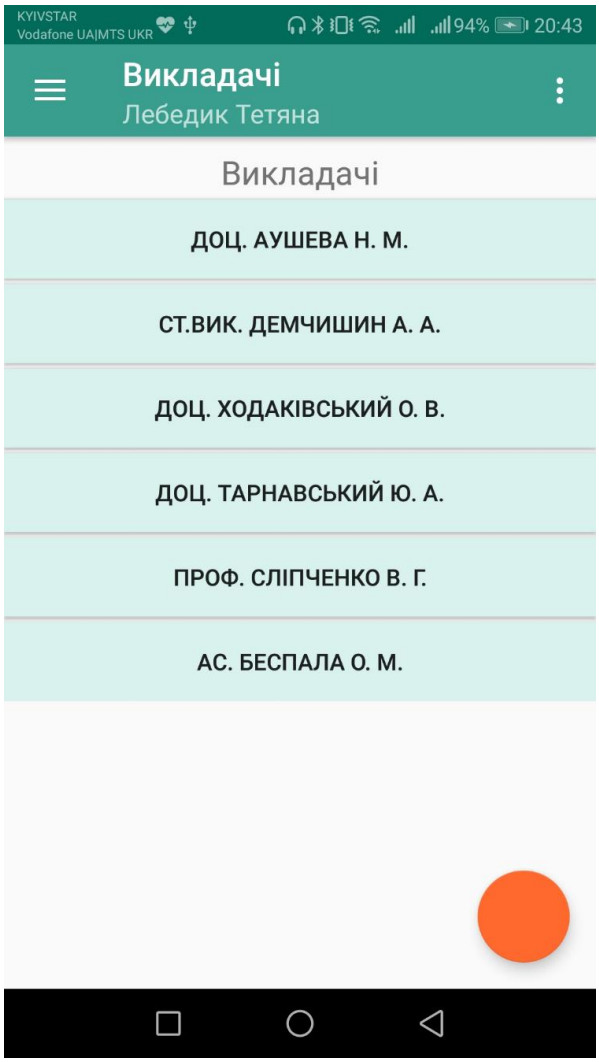

Рис 5.5 - Список викладачів

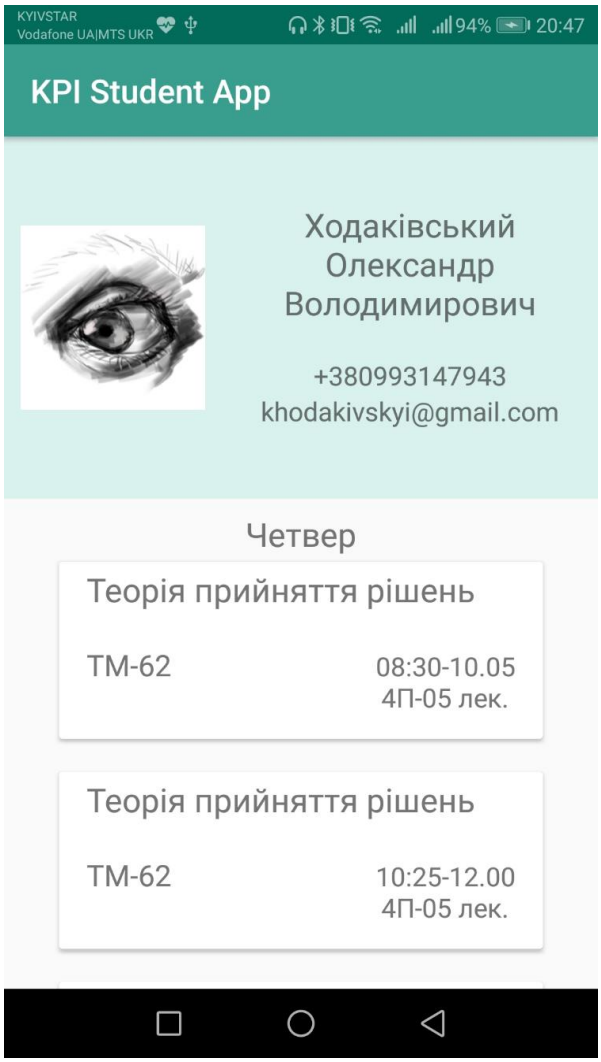

Рис 5.6 – Детальна інформація про викладачів

5.2 Мобільний додаток викладача

Має схожий інтерфейс та схему взаємодії з додатком студента. Після авторизації, поданої на рисунку 5.7, користувач бачить свій поточний розклад, та може комфортно слідкувати за ним. На рисунку 5.8 подано ситуацію, коли у викладача за розкладом, немає жодного заняття поточного дня.

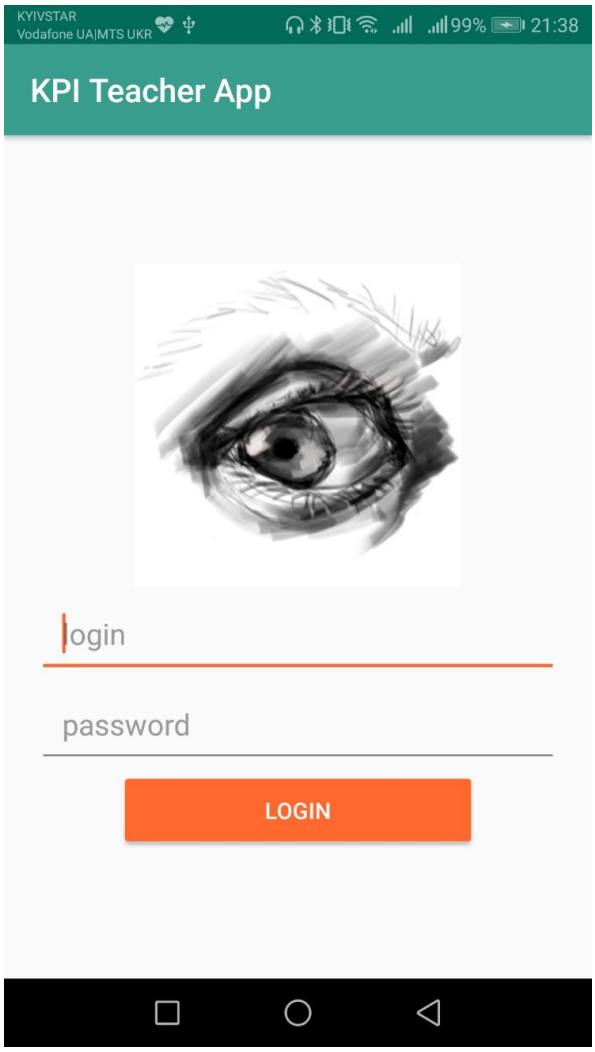

Рис 5.7 - Форма авторизації викладача

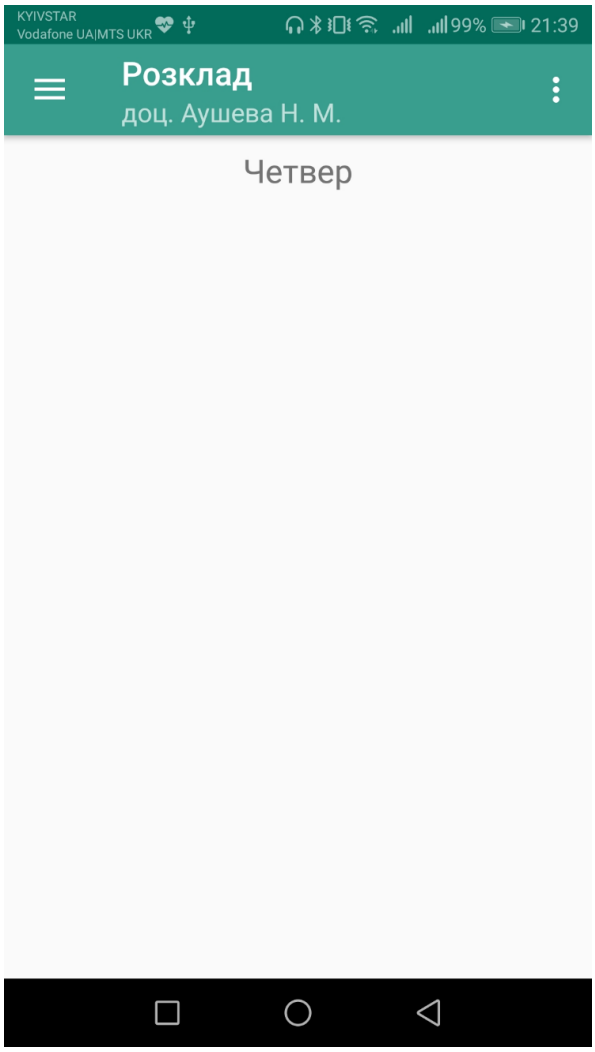

Рис. 5.8 - Розклад

Мобільний додаток містить такі вкладки, як: “Розклад”, “Студенти” та “Мапа”. Для переходу між вкладками потрібно натиснути на кнопку виклику бокового меню. Зявиться бокове меню, яке подано на рисунку 5.9.

На вкладці “Студенти”, викладач може обрати групу та конкретного студента для перегляду інформації про нього. Рисунки 5.10-5.11. Після натискання на ім’я студента, відкривається екран, на якому відображається детальна інформація про нього.

Додаток надає інформацію про ім’я та прізвище студента, його групу, курс, телефон та електронну адресу. На рисунку 5.12 показано детальну інформацію про студента, а також дати, коли студент був присутній на заняттях.

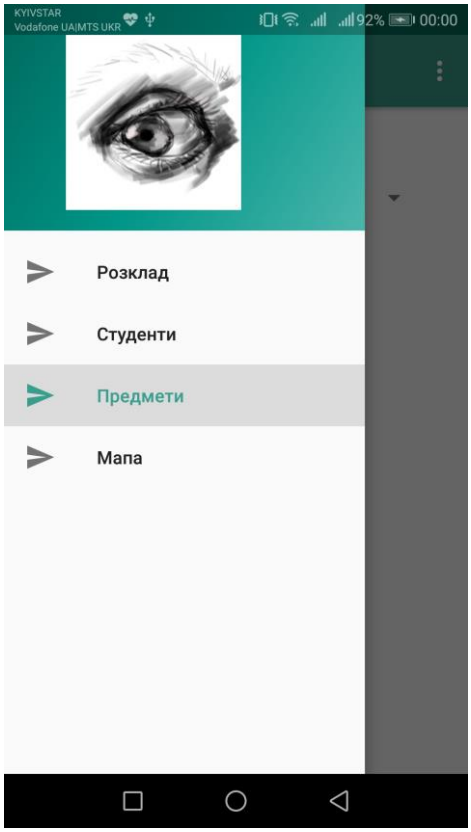

Рис 5.9 – Бокове меню переходу між вкладками

Рис 5.10 - Вкладка для вибору групи

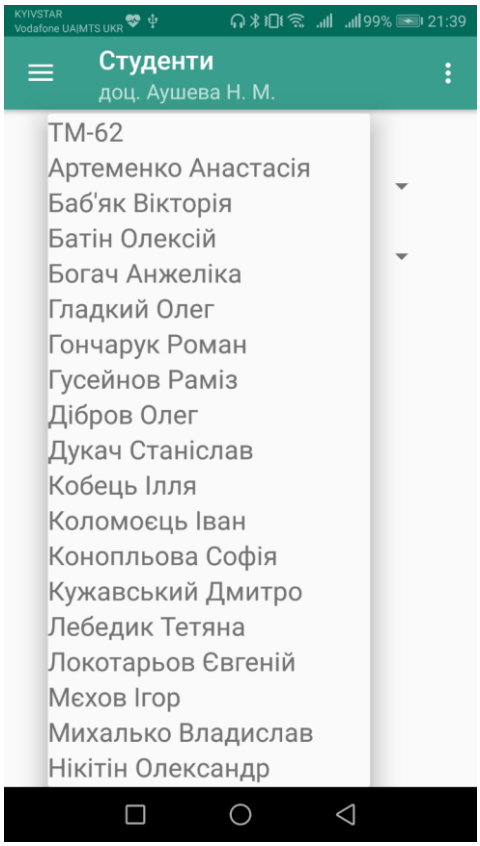

Рис 5.11 - Вибір студента

Рис 5.12 - Перегляд інформації про студента та його відвідування

Дати згруповано за типом заняття: лекція, лабораторна робота чи практичне заняття.

На рисунках 5.13-5.14 показано вкладку “Предмети”. Тут викладач може побачити список предметів, що він викладає. При натисканні на назву предмету можна переглянути список груп, які проходять даний курс у викладача.

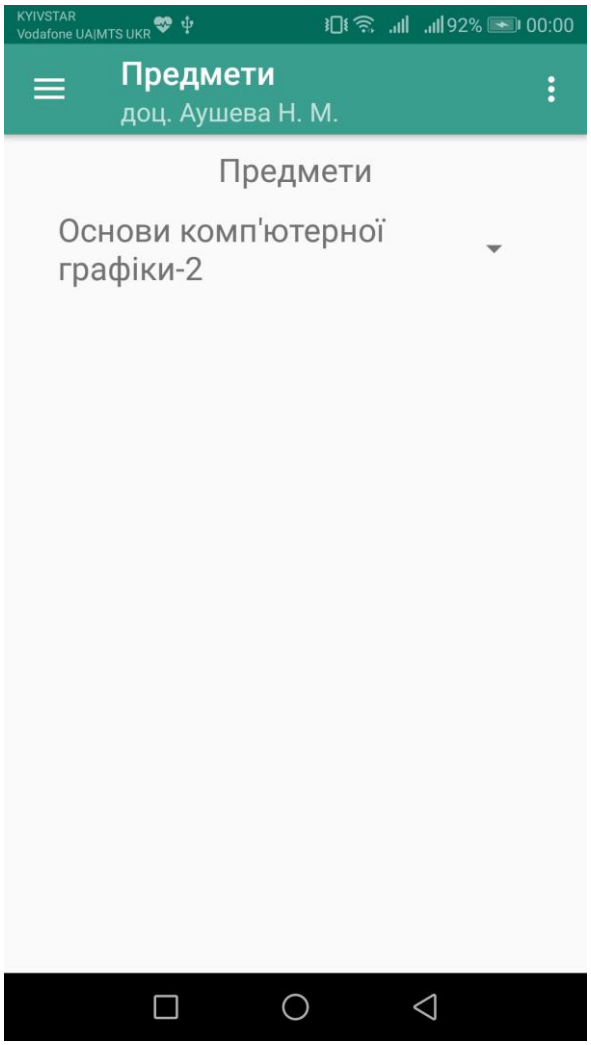

Рис 5.13 – Вибір предмету

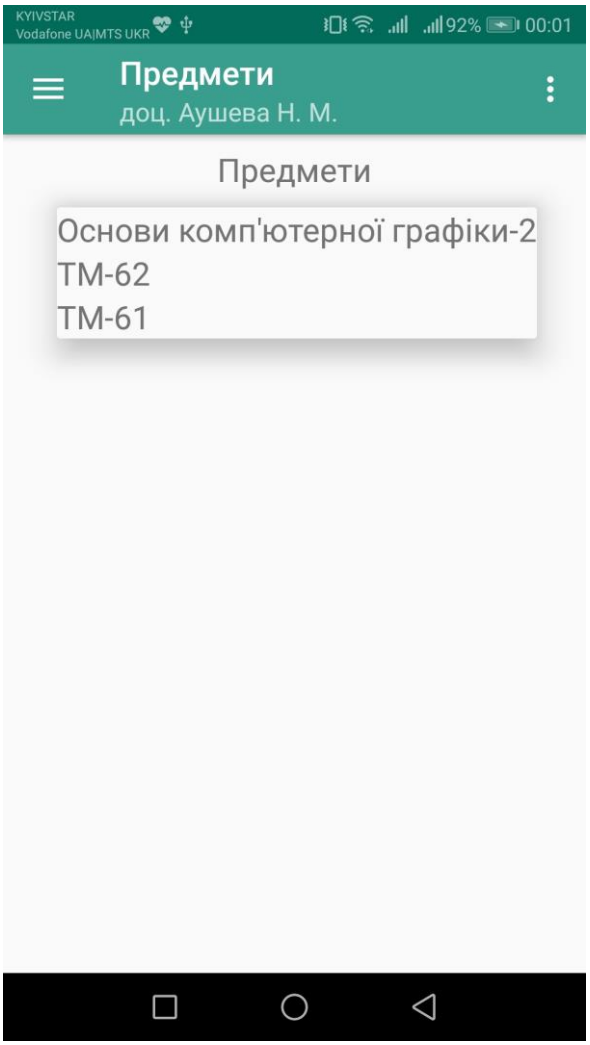

Рис 5.14 – Вибір групи

Після того, як викладач обере предмет та групу, йому буде показано успішність групи з даного предмету. Вкладку з відвідуванням показано на рисунках 5.15 – 5.16.

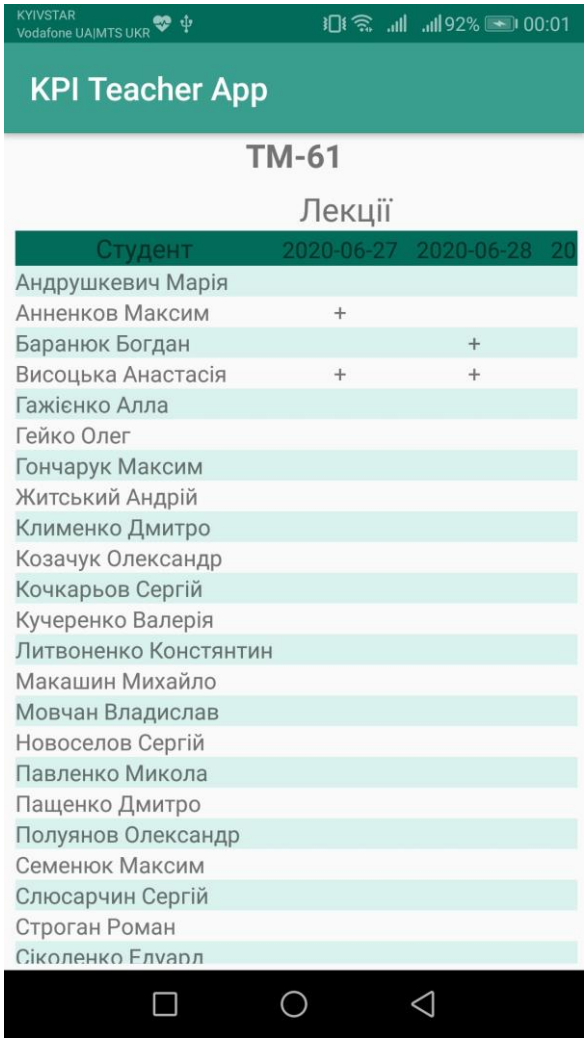

Рис 5.15 – Відвідування лекцій

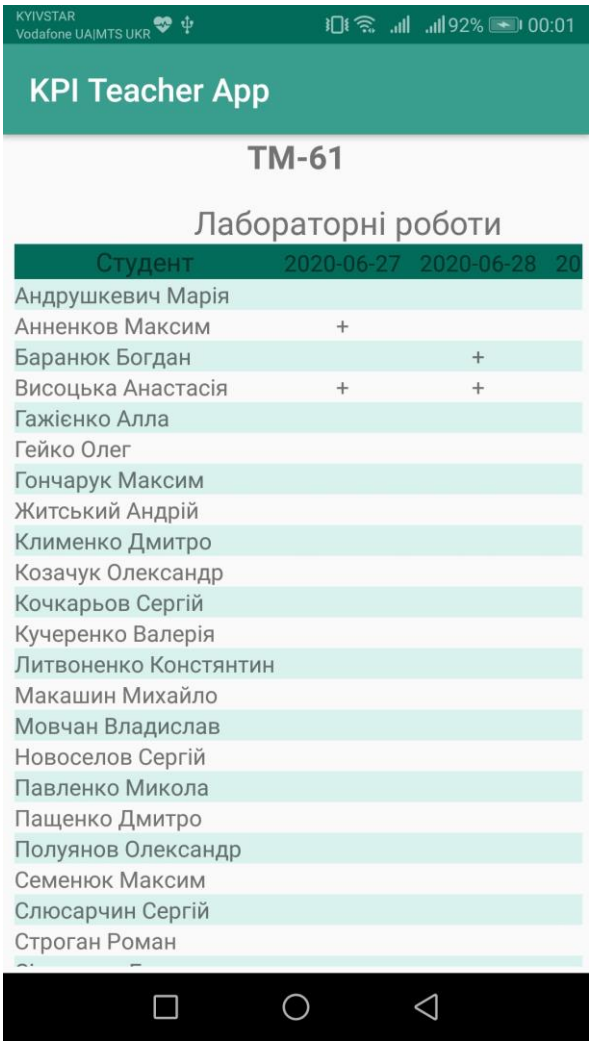

Рис 5.16 – Відвідування лабораторних робіт

Відвідування подано у вигляді таблиць. Першою колонкою є фамілії та імена студентів, далі подано дати проведення занять й відмітки про присутність кожного студента. Над таблицею подано тип заняття: лекція, лабораторна робота або практичне заняття.

5.3 Веб-додаток адміністратора.

Веб-додаток адміністратора бази даних призначено для редагування таблиць бази даних відвідування навчального закладу.

При вході до веб-додатку потрібно пройти авторизацію: ввести логін та пароль до форми. Форму авторизації подано на рисунку 5.17.

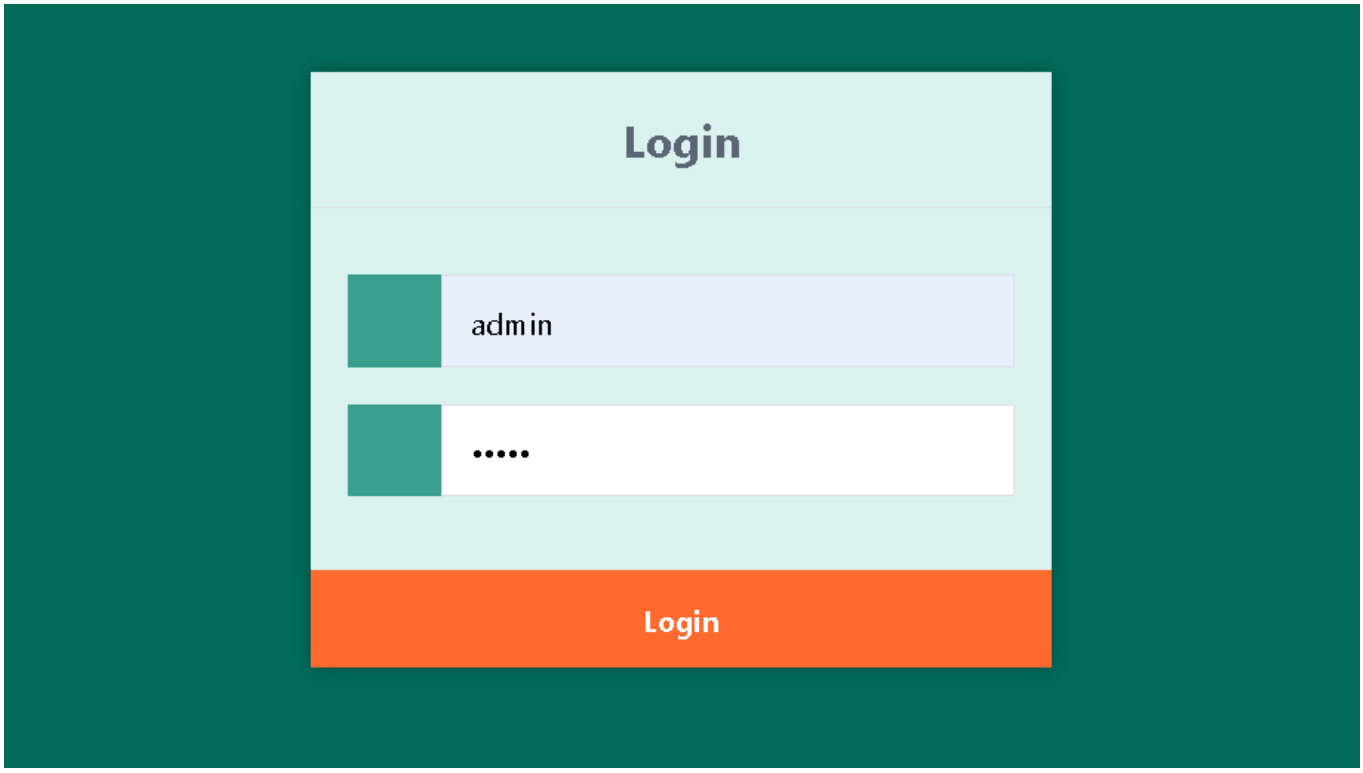

Рис 5.17 – Форма реєстрації адміністратора

Після проходження авторизації, користувач потрапляє на сторінку редагування таблиць(рисунок 5.18). З ліва представлено бокове меню, на якому можна обрати таблиці з бази даних відвідування, для редагування, а саме:

- “Бікони(Beacons)”,
- “Групи(Groups)”,
- “Студенти(Students)”,
- “Предмети(Subjects)”,
- “Викладачі(Teachers)”,
- “Заняття(Lessons)”,
- “Предмети у викладачів(Sub&Teach)”.

За замовчуванням відкривається таблиця редагування біконів навчального закладу - “Бікони(Beacons)”. Для виходу з додатку потрібно натиснути кнопку “Logout”.

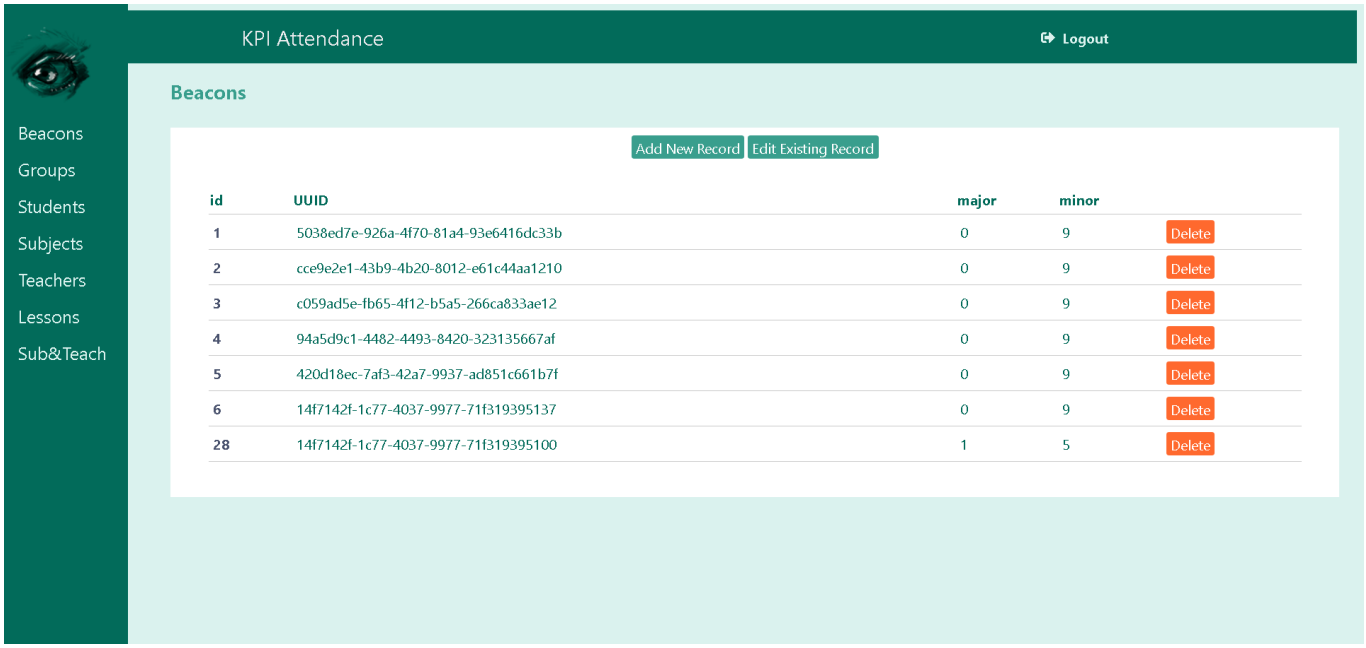

Рис 5.18 – Сторінка редагування таблиць

Додаток повністю реалізує CRUD операції для таблиць бази даних. Приклад вигляду таблиць “Студенти” та “Викладачі” з бази даних показано на рисунках 5.19-5.20.

Зверху кожної таблиці подано дві кнопки:

- “Add New Record” – при натисканні відкривається форма, для додавання нового запису до таблиці.
- “Edit Existing Record” - при натисканні відкривається форма, для редагування існуючого запису до таблиці.

В останній колонці, переліку значущих стовбців кожної таблиці наведено ще одну кнопку, для видалення запису:

- “Delete” – при натисканні видаляється поточний запис таблиці.

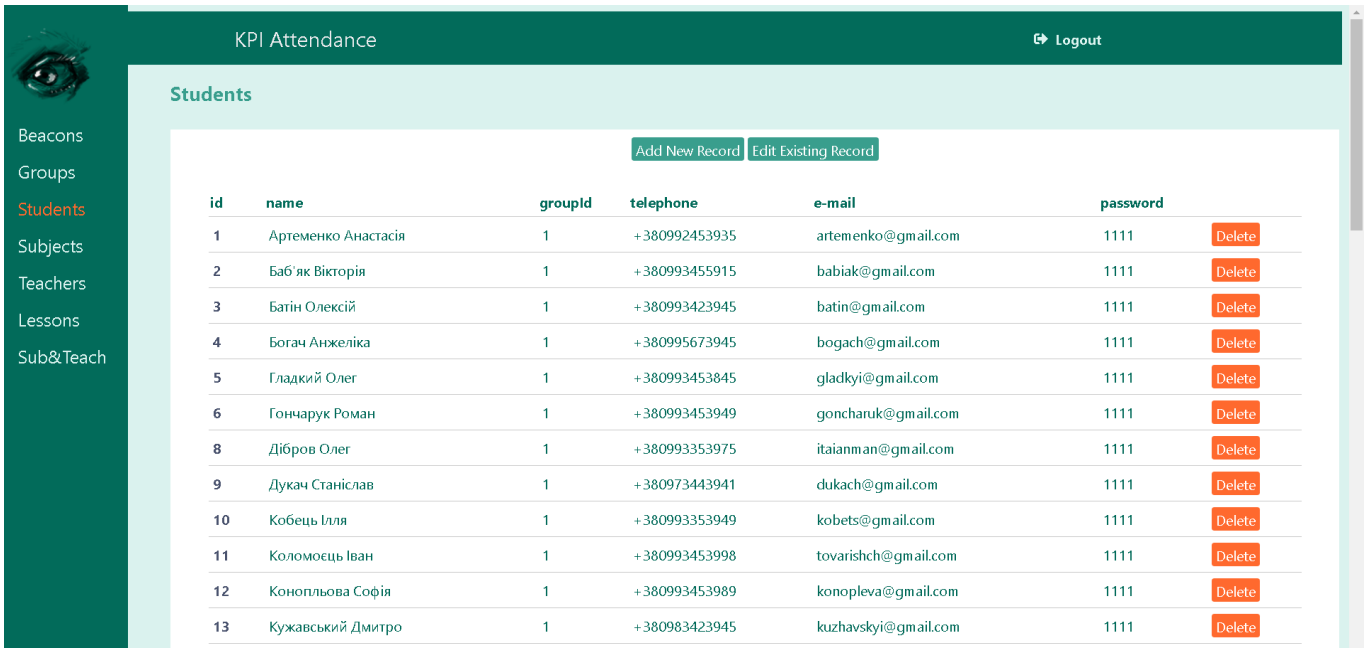

Рис 5.19 – Таблиця “Студенти”

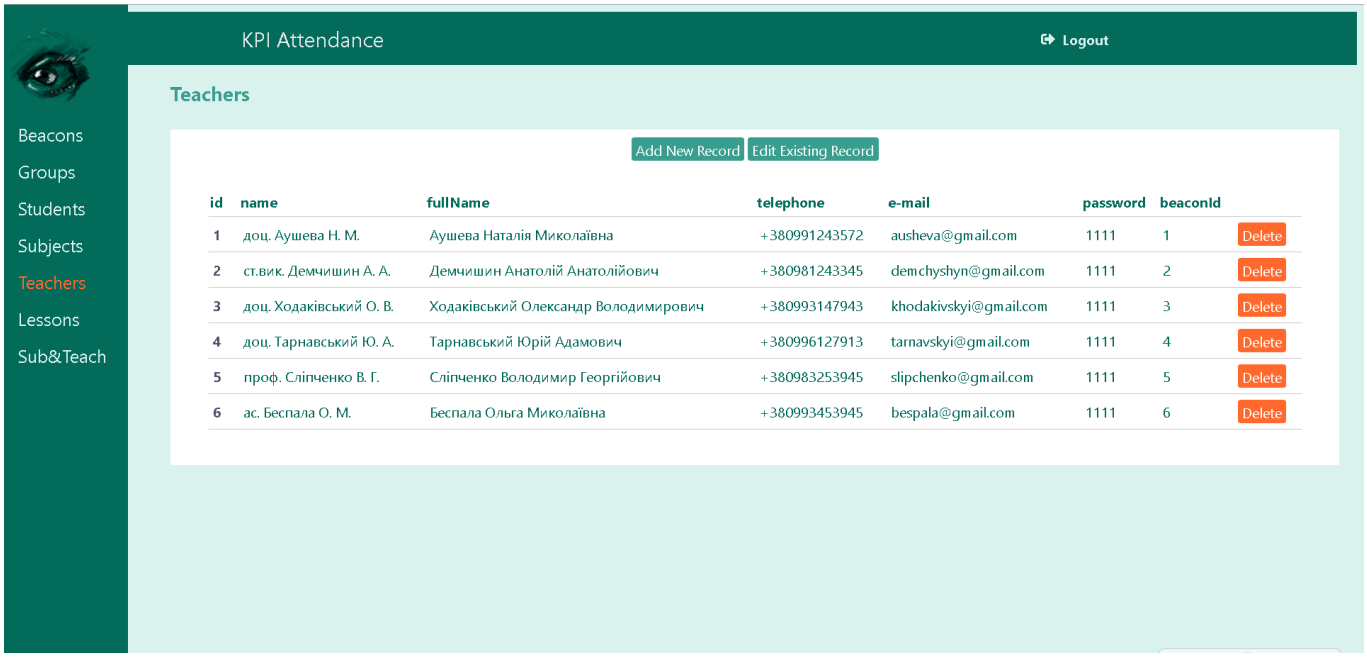

Рис 5.20 – Таблиця “Викладачі”

Форму додавання нового запису до таблиці бази даних подано на прикладі таблиці “Бікони” на рисунку 5.21.

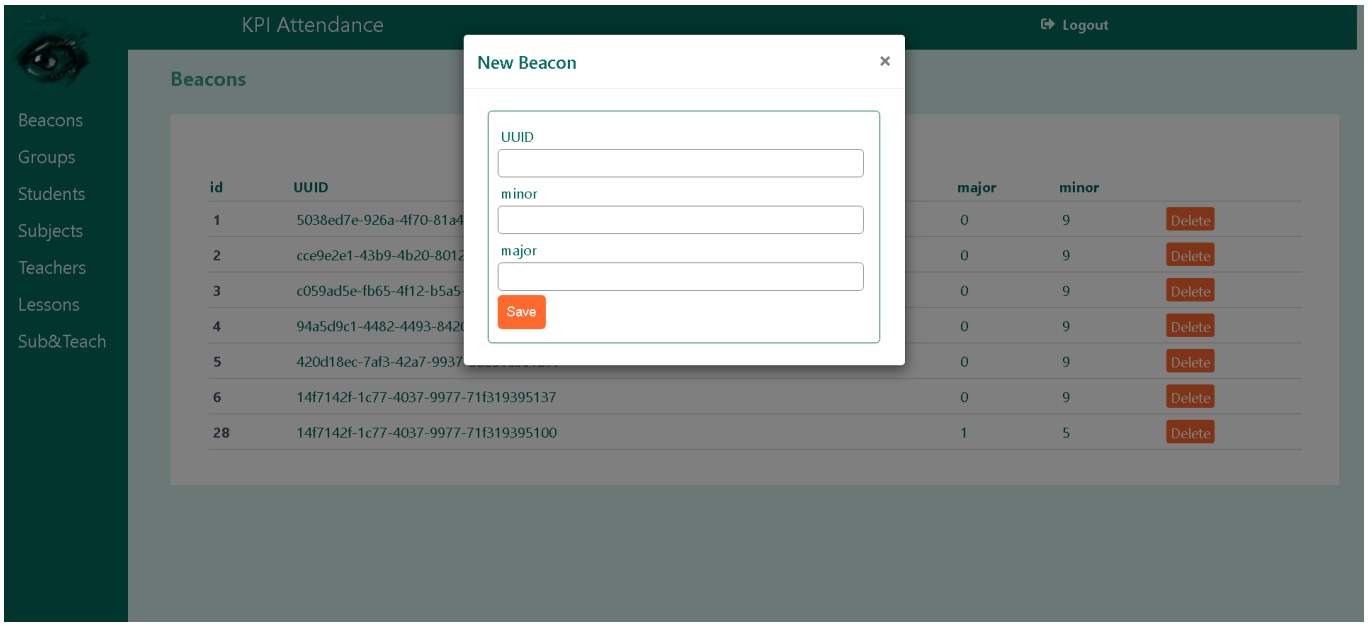

Рис 5.21 – Форма додавання нового запису.

Форму редагування існуючого запису до таблиці бази даних подано на прикладі таблиці “Бікони” на рисунку 5.22.

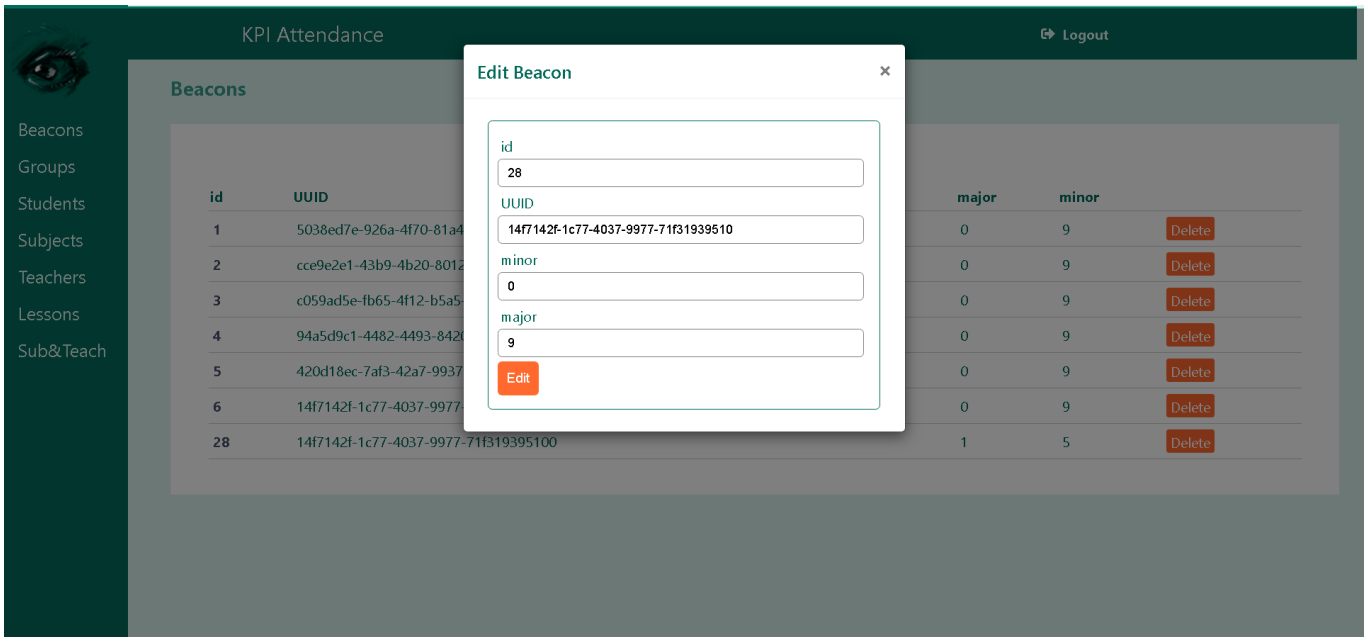

Рис 5.22 – Форма редагування існуючого запису.

ВИСНОВКИ

У дипломній роботі було вивчено спосіб внутрішньої локалізації на основі технології Bluetooth з низьким енергоспоживанням.

Було проведено аналіз існуючого програмного забезпечення, виявлено його позитивні та негативні характеристики, які враховано у процесі розробки програмного забезпечення.

Проведено огляд методів і засобів розробки програмної системи та обрано оптимальний набір технологій: мови програмування Kotlin, SQL та PHP, середовище розробки Intelijj IDEA Ultimate, система управління базами даних MySQL та система адміністрування PHP MyAdmin.

В ході виконання дипломної роботи було створено:

1. Мобільний додаток викладача, який надає можливість викладачу відслідковувати присутність студентів на заняттях.
2. Мобільний додаток студента, який надає можливість студентам підтвердити свою присутність на заняттях.
3. Базу даних для збереження інформації про розклад, викладачів, студентів та їх відвідування.
4. Веб-додаток для адміністрування бази даних.

Таким чином, було виконано основну мету дипломної роботи, створено програмне забезпечення, яке вирішує проблему обліку відвідування навчального закладу, на основі технології біконів.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Faragher R. Location Fingerprinting With Bluetooth Low Energy Beacons / R. Faragher, R. Harle., 2015. – 2418 с.

2. Gu Y. Energy-Efficient Indoor Localization of Smart Hand-Held Devices Using Bluetooth / Y. Gu, F. Ren., 2015. – 1450 с.

3. Mark up the world using beacons [Електронний ресурс] // Google – Режим доступу до ресурсу: https://developers.google.com/beacons/.

4. DeCuir J. Introducing Bluetooth Smart: Part 1: A look at both classic and new technologies / Joseph DeCuir., 2013. – 12 с.

5. A stigmergic approach to indoor localization using Bluetooth Low Energy beacons / F.Palumbo, P. Barsocchi, S. Chessa, J. Augusto., 2015.

6. Free Transparent PNG Images [Електронний ресурс] – Режим доступу до ресурсу: https://favpng.com/.

7. Mokosmart [Електронний ресурс] – Режим доступу до ресурсу: https://www.mokosmart.com.

8. iBeacon [Електронний ресурс] // Apple – Режим доступу до ресурсу: https://developer.apple.com/ibeacon/.

9. AltBeacon [Електронний ресурс] // AltBeacon – Режим доступу до ресурсу: https://altbeacon.org/.

10. MDPI [Електронний ресурс] – Режим доступу до ресурсу: mdpi.com.

11. Specification for Eddystone, an open beacon format from Google [Електронний ресурс] // Google – Режим доступу до ресурсу: https://github.com/google/eddystone.

12. Hong Kong International Airport and SITA trial beacons for better passenger experience [Електронний ресурс] – Режим доступу до ресурсу: https://www.sita.aero/pressroom/news-releases/hong-kong-international-airport-and-sita-trial-beacons-for-better-passenger-experience.

13. The Smartphone Revolution Was the Android Revolution [Електронний ресурс]. – 2019. – Режим доступу до ресурсу: https://www.bloomberg.com/graphics/2019-android-global-smartphone-growth/.

14. Дарвин Я. Android. Сборник рецептов: задачи и решения для разработчиков приложений / Ян Ф. Дарвин., 2019. – (768).

15. Скин Д. Kotlin. Программирование для профессионалов / Д. Скин, Д. Гринхол.. – 464 с. – (Питер).

16. IntelliJ IDEA [Електронний ресурс] – Режим доступу до ресурсу: https://www.jetbrains.com/ru-ru/idea/.

17. MySQL [Електронний ресурс] // Oracle Corporation – Режим доступу до ресурсу: https://www.mysql.com/.

18. Bringing MySQL to the web [Електронний ресурс] – Режим доступу до ресурсу: https://www.phpmyadmin.net/.

19. PhpStorm [Електронний ресурс] – Режим доступу до ресурсу: https://www.jetbrains.com/ru-ru/phpstorm/.

20. JetBrains [Електронний ресурс] // JetBrains s.r.o. – Режим доступу до ресурсу: https://www.jetbrains.com/.

21. Gradle Build Tool [Електронний ресурс] // Gradle Inc. – Режим доступу до ресурсу: https://gradle.org/.

22. A Java serialization/deserialization library to convert Java Objects into JSON and back [Електронний ресурс] – Режим доступу до ресурсу: https://github.com/google/gson.

23. Android Jetpack Compose [Електронний ресурс] // Android – Режим доступу до ресурсу: https://developer.android.com/jetpack/compose.

ДОДАТОК 1

Інструментальні засоби локалізації об’єктів на основі технології біконів

Специфікація

УКР.НТУУ”КПІ ім. Ігоря Сікорського”_ТЕФ_АПЕПС_ТМ62202_20Б 81-1

Аркушів 2

Київ – 2020

Позначення	Найменування	Примітки
Документація		
УКР.НТУУ«КПІ ім. Ігоря Сікорського»_ТЕФ_АПЕПС_ТМ 62202_20Б 81-1	Записка.docx	Пояснювальна записка
Компоненти		
УКР.НТУУ«КПІ ім. Ігоря Сікорського»_ТЕФ_АПЕПС_ТМ 62202_20Б 12-1	KPIStudentApp\app\src\main\java\com\attendence\kpistudentapp\NavigationDrawerActivity.kt	Модуль роботи з біконами, додатку студента
УКР.НТУУ«КПІ ім. Ігоря Сікорського»_ТЕФ_АПЕПС_ТМ 62202_20Б 12-2	KPIStudentApp\app\src\main\java\com\attendence\kpistudentapp\ui\schedule\ScheduleFragment.kt	Модуль розкладу, який безпосередньо впливає на роботу BLE
УКР.НТУУ«КПІ ім. Ігоря Сікорського»_ТЕФ_АПЕПС_ТМ 62202_20Б 12-3	KPIStudentApp\app\src\main\res\layout\fragment_schedule.xml	Візуальна форма модулю розкладу
УКР.НТУУ”КПІ ім. Ігоря Сікорського”_ТЕФ_АПЕПС_ТМ 62202_20 13-1	Опис.docx	Опис програми

ДОДАТОК 2

Інструментальні засоби локалізації об’єктів на основі технології біконів

Текст програми

УКР.НТУУ”КПІ ім. Ігоря Сікорського”_ТЕФ_АПЕПС_ТМ62202_20Б 12-1

Аркушів 8

Київ – 2020

УКР.НТУУ«КПІ ім. Ігоря Сікорського»_ТЕФ_АПЕПС_ТМ62202_20Б 12-1

// NavigationDrawerActivity.activity

```
package com.attendence.kpistudentapp

import android.Manifest
import android.content.pm.PackageManager
import android.os.Build
import android.os.Bundle
import android.os.RemoteException
import android.view.Menu
import androidx.appcompat.app.AlertDialog
import androidx.appcompat.app.AppCompatActivity
import androidx.appcompat.widget.Toolbar
import androidx.drawerlayout.widget.DrawerLayout
import androidx.navigation.findNavController
import androidx.navigation.ui.AppBarConfiguration
import androidx.navigation.ui.navigateUp
import androidx.navigation.ui.setupActionBarWithNavController
import androidx.navigation.ui.setupWithNavController
import com.attendence.kpistudentapp.db.attendance.DateIfRecordExist
import com.attendence.kpistudentapp.db.attendance.InsertAttendance
import com.attendence.kpistudentapp.db.beacon.BeaconByTeacher
import com.attendence.kpistudentapp.db.group.GroupNameByStudent
import com.attendence.kpistudentapp.db.lessons.LessonForStudent
import com.attendence.kpistudentapp.db.lessons.LoadAllStudentsLessons
import com.attendence.kpistudentapp.db.student.LoadStudentId
import com.attendence.kpistudentapp.db.student.LoadStudentName
import com.google.android.material.floatingactionbutton.FloatingActionButton
import com.google.android.material.navigation.NavigationView
import com.google.android.material.snackbar.Snackbar
import org.altbeacon.beacon.*
import java.time.LocalDate
import java.time.LocalTime
import java.util.*

class NavigationDrawerActivity : AppCompatActivity(), BeaconConsumer {
```

```
    private lateinit var appBarConfiguration: AppBarConfiguration
    private val PERMISSION_REQUEST_COARSE_LOCATION = 1
    private var beaconManager:BeaconManager? = null
    private var UUID: String? = null

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_navigation_drawer)
        val toolbar: Toolbar = findViewById(R.id.toolbar)



        val intent = intent
        val message = intent.extras?.getString("student")

        val loadStudentName = LoadStudentName(this)
        val loadStudentNameExecute = loadStudentName.execute(message)
        val student_string = loadStudentNameExecute.get()

        toolbar.setSubtitle(student_string);
        setSupportActionBar(toolbar)

        val drawerLayout: DrawerLayout = findViewById(R.id.drawer_layout)
        val navView: NavigationView = findViewById(R.id.nav_view)
        val navController = findNavController(R.id.nav_host_fragment)

        appBarConfiguration = AppBarConfiguration(
            setOf(
                R.id.nav_schedule, R.id.nav_students, R.id.nav_map
            ), drawerLayout
        )
        setupActionBarWithNavController(navController, appBarConfiguration)
        navView.setupWithNavController(navController)

        if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.M) {
            // Android M Permission check
            if (this.checkSelfPermission(Manifest.permission.ACCESS_COARSE_LOCATION) !=
PackageManager.PERMISSION_GRANTED) {
                val builder = AlertDialog.Builder(this)
                builder.setTitle("This app needs location access")
                builder.setMessage("Please grant location access so this app can detect
```

```
beacons in the background.")
                builder.setPositiveButton(android.R.string.ok, null)
                builder.setOnDismissListener {
                    requestPermissions(arrayOf(Manifest.permission.ACCESS_COARSE_LOCATION),
                        PERMISSION_REQUEST_COARSE_LOCATION)
                }
                builder.show()
            }
        }

        beaconManager = BeaconManager.getInstanceForApplication(this)
        beaconManager!!.getBeaconParsers().add(
            BeaconParser().setBeaconLayout(BeaconParser.EDDYSTONE_UID_LAYOUT)
        );
        beaconManager!!.bind(this)
        beaconManager!!.setMaxTrackingAge(5)
        onBeaconServiceConnect()

val c = Calendar.getInstance()
            c.time = Date()
            val dayOfWeek = c[Calendar.DAY_OF_WEEK]

            val loadStudentId = LoadStudentId(this)
            val loadStudentIdExecute = loadStudentId.execute(toolbar.subtitle.toString())
            val student_id: String = loadStudentId.get()

            val loadGroup = GroupNameByStudent(this)
            val loadGroupExecute = loadGroup.execute(toolbar.subtitle.toString())
            val group_name: String = loadGroupExecute.get()
            val loadAllStudentsLessons = LoadAllStudentsLessons(this)
            val loadAllStudentsLessonsExecute =
loadAllStudentsLessons.execute(dayOfWeek.toString(), "1", group_name)
            val lessons: List<LessonForStudent>? = loadAllStudentsLessonsExecute.get()
            val final_lessons = mutableListOf<LessonForStudent>()
            if (lessons != null) {
                 for(i in lessons!!){
                      final_lessons.add(i)
                  }

            var is_record_insert: Boolean = false
```

```
            for(i in final_lessons){
                val loadBeacon = BeaconByTeacher(this)
                val loadBeaconExecute = loadBeacon.execute(i.teacher)
                val beaconUUID:String = loadBeaconExecute.get()
                if(isCurrentLesson(i.lesson_number_time) && beaconUUID == UUID){
                    val date = LocalDate.now()

                    val recordExist = DateIfRecordExist(this)
                    val recordExistExecute = recordExist.execute(student_id,
i.id.toString(), date.toString())
                    val check: String = recordExistExecute.get()
                    if(check != date.toString()){
                        val insertAttendance = InsertAttendance(this)
                        val loadAttendanceExecute =
insertAttendance.execute(i.id.toString(), student_id, date.toString())
                        val insert_result:String = loadAttendanceExecute.get()
                        if(insert_result == "New record created successfully"){
                            is_record_insert = true
                        }
                    }

                }
            }
        }
    }

    override fun onCreateOptionsMenu(menu: Menu): Boolean {
        menuInflater.inflate(R.menu.navigation_drawer, menu)
        return true
    }

    override fun onSupportNavigateUp(): Boolean {
        val navController = findNavController(R.id.nav_host_fragment)
        return navController.navigateUp(appBarConfiguration) || super.onSupportNavigateUp()
    }

    override fun onBeaconServiceConnect() {
        beaconManager!!.setRangeNotifier(object : RangeNotifier {
            override fun didRangeBeaconsInRegion(beacons: Collection<Beacon>, region:
Region) {
```

```
                if (beacons.size > 0) {

                    val firstBeacon = beacons.iterator().next()
                    runOnUiThread {
                        println("The coffee beacon " + firstBeacon.toString() + " is about "
+ firstBeacon.distance + " meters away.")
                        UUID = firstBeacon.toString()
                    }
                }
            }
        })
        try {
            beaconManager!!.startRangingBeaconsInRegion(Region("myRangingUniqueId", null,
null, null))
        } catch (e: RemoteException) {
        }

    }

    fun onBeaconServiceDissconnect(){
        beaconManager!!.unbind(this);
    }

}
```


Програмний код класу ScaduleFragment, який є одним із компонентів класу NavigationDrawerActivity і відповідає за відображення поточного розкладу студента:

```
package com.attendence.kpistudentapp.ui.schedule

import android.os.Bundle
import android.util.TypedValue
import android.view.Gravity
import android.view.LayoutInflater
import android.view.View
import android.view.ViewGroup
import android.widget.LinearLayout
import android.widget.TextView
import androidx.appcompat.widget.Toolbar
import androidx.cardview.widget.CardView
```

```
import androidx.core.content.ContextCompat
import androidx.fragment.app.Fragment
import com.attendence.kpistudentapp.R
import com.attendence.kpistudentapp.TextConstants
import com.attendence.kpistudentapp.db.group.GroupNameByStudent
import com.attendence.kpistudentapp.db.lessons.LessonForStudent
import com.attendence.kpistudentapp.db.lessons.LoadAllStudentsLessons
import java.time.LocalTime

import java.util.*


class ScheduleFragment : Fragment() {

    override fun onCreateView(inflater: LayoutInflater, container: ViewGroup?,
savedInstanceState: Bundle?): View? {

        val root = inflater.inflate(R.layout.fragment_schedule, container, false)

        val dayMonthText: TextView = root.findViewById(R.id.week_and_day_text)
        val c = Calendar.getInstance()
        c.time = Date()
        val dayOfWeek = c[Calendar.DAY_OF_WEEK]
        dayMonthText.text = TextConstants().getUkrDayName(dayOfWeek)

        val cardLinearLayout: LinearLayout = root.findViewById(R.id.card_linear_layout)
        val toolbar: Toolbar = requireActivity().findViewById(R.id.toolbar)


        val loadGroup = GroupNameByStudent(container!!.context)
        val loadGroupExecute = loadGroup.execute(toolbar.subtitle.toString())
        val group_name:String = loadGroupExecute.get()


        val loadAllStudentsLessons = LoadAllStudentsLessons(container!!.context)
        val loadAllStudentsLessonsExecute =
loadAllStudentsLessons.execute(dayOfWeek.toString(), "1", group_name)
        val lessons:List<LessonForStudent>? = loadAllStudentsLessonsExecute.get()
        val final_lessons = mutableListOf<LessonForStudent>()
        if(lessons != null) {
```

```
            for (i in lessons!!) {
                final_lessons.add(i)
            }
            final_lessons.sortBy { it.lesson_number_time }
            for (i in final_lessons) {
                cardLinearLayout.addView(getCard(container, i))
            }
        }
        return root
    }

    private fun getCard(container: ViewGroup?, lesson:LessonForStudent): CardView {

        val layoutParams = LinearLayout.LayoutParams(
            LinearLayout.LayoutParams.MATCH_PARENT, // CardView width
            LinearLayout.LayoutParams.WRAP_CONTENT // CardView height
        )
        layoutParams.setMargins(100, 10, 100, 50)

        val lessonCardView = CardView(container!!.context)
        lessonCardView.layoutParams = layoutParams
        val verticalLinearLayout = LinearLayout(container!!.context)
        verticalLinearLayout.orientation = LinearLayout.VERTICAL

        if(isCurrentLesson(lesson.lesson_number_time)){

lessonCardView.setCardBackgroundColor(ContextCompat.getColor(container!!.context,
R.color.colorAccent))
        }

        val horizontalLinearLayout = LinearLayout(container!!.context)
        horizontalLinearLayout.addView(getTeacher(container, lesson))
        horizontalLinearLayout.addView(getLocation(container, lesson))

        verticalLinearLayout.addView(getSubject(container, lesson))
        verticalLinearLayout.addView(horizontalLinearLayout)

        lessonCardView.addView(verticalLinearLayout)

        return lessonCardView
```

```
}

private fun getSubject(container: ViewGroup?, lesson:LessonForStudent): TextView{

    val subjectTextLayoutParams = LinearLayout.LayoutParams(
        LinearLayout.LayoutParams.MATCH_PARENT,
        LinearLayout.LayoutParams.WRAP_CONTENT
    )
    subjectTextLayoutParams.setMargins(50, 10, 50, 50)

    val subjectNameTextView = TextView(container!!.context)
    subjectNameTextView.setText(lesson.name)
    subjectNameTextView.layoutParams = subjectTextLayoutParams
    subjectNameTextView.setTextSize(TypedValue.COMPLEX_UNIT_SP, 20f)

    return subjectNameTextView
}

private fun getTeacher(container: ViewGroup?, lesson:LessonForStudent): TextView {

    val teacherTextLayoutParams = LinearLayout.LayoutParams(
        LinearLayout.LayoutParams.MATCH_PARENT,
        LinearLayout.LayoutParams.WRAP_CONTENT,
        1f
    )
    teacherTextLayoutParams.setMargins(50, 10, 50, 50)

    val teacherNameTextView = TextView(container!!.context)
    teacherNameTextView.layoutParams = teacherTextLayoutParams
    teacherNameTextView.setText(lesson.teacher)
    teacherNameTextView.setTextSize(TypedValue.COMPLEX_UNIT_SP, 16f)
    teacherNameTextView.gravity = Gravity.BOTTOM

    return teacherNameTextView
}

private fun getLocation(container: ViewGroup?, lesson:LessonForStudent): TextView{

    val locationTextLayoutParams = LinearLayout.LayoutParams(
        LinearLayout.LayoutParams.MATCH_PARENT,
```

```
            LinearLayout.LayoutParams.WRAP_CONTENT,
            1f
        )
        locationTextLayoutParams.setMargins(50, 10, 50, 50)

        val timeLocationTextView = TextView(container!!.context)
        timeLocationTextView.layoutParams = locationTextLayoutParams

timeLocationTextView.setText(TextConstants().getLessonTime(lesson.lesson_number_time)+"\n" +
lesson.room + " " + lesson.type)
        timeLocationTextView.setTextSize(TypedValue.COMPLEX_UNIT_SP, 16f)
        timeLocationTextView.gravity = Gravity.END
        //timeLocationTextView.gravity = Gravity.RIGHT

        return timeLocationTextView
    }
}
```


Xml файл, що відповідає за відображення розкладу та керується Kotlin класом ScaduleFragment:

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
        xmlns:android="http://schemas.android.com/apk/res/android"
        xmlns:app="http://schemas.android.com/apk/res-auto"
        android:layout_width="match_parent"
        android:layout_height="match_parent">

    <LinearLayout
            android:orientation="vertical"
            android:layout_width="match_parent"
            android:layout_height="match_parent">
        <TextView
                android:id="@+id/week_and_day_text"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:layout_marginStart="8dp"
                android:layout_marginTop="8dp"
                android:layout_marginEnd="8dp"
                android:textAlignment="center"
```

```
            android:textSize="20sp"
        />
        <ScrollView
            android:layout_width="match_parent"
            android:layout_height="match_parent">
            <LinearLayout
                android:id="@+id/card_linear_layout"
                android:orientation="vertical"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"/>
        </ScrollView>
    </LinearLayout>

</androidx.constraintlayout.widget.ConstraintLayout>
```

ДОДАТОК 3

Інструментальні засоби локалізації об’єктів на основі технології біконів

Опис програми

УКР.НТУУ”КПІ ім. Ігоря Сікорського”_ТЕФ_АПЕПС_ТМ62202_20Б 13-1

Аркушів 8

Київ – 2020

АНОТАЦІЯ

Програмна система надає можливість проводити облік відвідування навчального закладу. Складається з бази даних, веб-додатку адміністратора та двох мобільних додатків для студента та викладача.

Додаток студента надає інформацію про розклад та викладачів, а також працює як сканер бікнів, для локалізації присутності на занятті. Додаток викладача надає йому інформацію про розклад, студентів та їх відвідування.

Мобільні додатки створено з використанням мови програмування Kotlin у середовищі розробки IntelliJIDEA Ultimate. Для створення бази даних, та налагодження звязку з нею мобільних додатків використано мови SQL та PHP.

ЗМІСТ

-4-

ЗАГАЛЬНІ ВІДОМОСТІ

Розроблена програмна система складається з двох мобільних додатків, веб-додатку адміністратора та бази даних.

Система дозволяє вирішити проблему обліку відвідування навчального закладу. Для вирішення проблеми використовуються спеціальні пристрої бікони, які транслюють сигнал. Мобільний додаток студента виступає в ролі сканера та вміє читати ці сигнали, таким чином здійснюючи локалізацію на місцевості.

Мобільні додатки створено на базі операційної системи Android, тому для їх використання потрібен мобільний пристрій з операційною системою Android.

Додатки по потрібно встановити на мобільний пристрій, при чому для встановлення знання програмування не потрібно.

-5-

ФУНКЦІОНАЛЬНЕ ПРИЗНАЧЕННЯ

Розроблена програмна система вирішує проблему обліку відвідування навчального закладу, на основі використання BLE технології. Система вирішує проблему шляхом локалізації студента на занятті.

Мобільний пристрій працює як сканер сигналу бікону викладача, а отже може однозначно локалізувати його місцезнаходження.

-6-

ОПИС ЛОГІЧНОЇ СТРУКТУРИ

Розроблена програмна система складається з двох мобільних додатків, веб-додатку адміністратора та бази даних, алгоритм роботи системи наступний:

1. Кожен викладач навчального закладу, у якому впроваджена система, має особистий бікон з унікальним ідентифікатором, який транслює сигнал для студентів, що присутні в аудиторії.

2. Кожен студент має змогу ввімкнути мобільний пристрій, на якому встановлено додаток для відслідковування відвідування, зайти до нього та авторизуватися. Після чого він отримує доступ до поточного розкладу та може переглянути список поточних занять.

3. Додаток студента фіксує сигнал, який транслюється біконом викладача й у певний час, за розкладом, студент має можливість поставити відмітку про свою присутність в аудиторії, для цього, йому потрібно лише ввімкнути мобільний додаток, який автоматично відправить інформацію про його присутність до бази даних.

4. Викладач має можливість переглянути відмітки про присутність студентів.

5. Редагуванням бази даних відвідування навчального закладу, реєстрація нових викладачів, студентів, предметів та біконів, які використовує кожен викладач, здійснюється адміністратором бази даних, за допомогою веб додатку для адміністрування.

-7-

ВИКОРИСТОВУВАНІ ТЕХНІЧНІ ЗАСОБИ

Для використання програмної системи потрібно мати:

Мобільний пристрій, що базується на операційній системі Android, на який встановлюється додаток викладача або додаток студента.

Доступ до будь-якого веб-браузера, для використання веб-додатку адміністратора.

-8-

ВХІДНІ І ВИХІДНІ ДАНІ

Вхідна інформація:

- розклад
- дані про викладачів
- дані про студентів
- дані про бікони

Вихідна інформація:

- таблиці відвідування